

المبادئ الأساسية للبرمجة بلغة

Delphi 6.0

مهندس

شريف فتحى الشافعى

المبادئ الأساسية للبرمجة
بلغة
Delphi 6.0



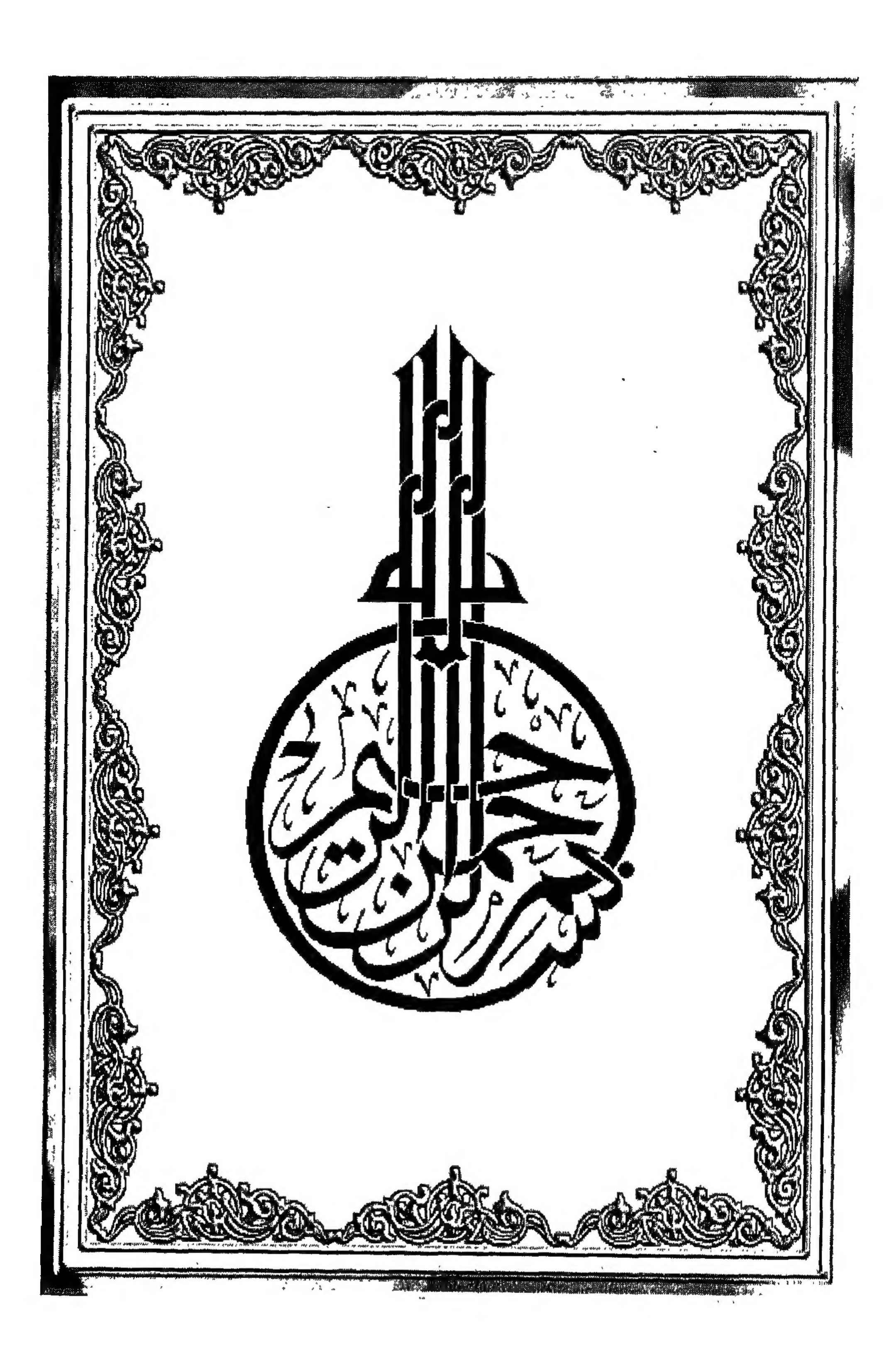
مهندس
شريف فتحى الشافعى

٢٠٠٣

رقـم الإيـداع بدار الكتب : ٢٠٠٢/١٧٥٢٨
الترقيـم الدولـي : ٩٧٧-٢٨٧-٢٧٧-٣

دار الكتب العلمية للنشر والتوزيع
٥٠ شارع الشيخ ريحان - عابدين - القاهرة
☎ ٧٩٥٤٢٢٩

لمزيد من المعلومات يرجى زيارة موقعنا على الأنترنت
www.scientificbookhouse.com
e-mail: sbh@link.net


بسم الله الرحمن الرحيم

الإهداء

إلى زوجتى الحبيبة الغالية :

التى كان لها فضل كبير فى خروج هذا الكتاب إلى النور.

الفصل الأول

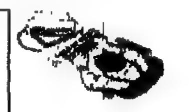
الجديد فى
Delphi 6



ما هو الجديد فى Delphi 6 ؟

من تعامل مع الإصدارات السابقة للغة البرمجة Delphi ثم أصبح الآن يتعامل مع الإصدار السادس لهذه اللغة فسيجد أن Delphi 6 تعمل على تقديم العديد من المظاهر والإمكانيات كما إنها تشتمل على كم هائل من التحسينات. وهذه الإمكانيات والتحسينات تتركز فى المناطق التالية :

- بيئة العمل IDE (اختصار للمصطلح Integrated Development Environment والذى يعنى بيئة التطوير المتكاملة).
- إعداد تطبيقات الإنترنت.
- التفاعل مع لغة البرمجة XML.
- المترجم Compiler الخاص باللغة.
- التعامل مع أدوات التحكم ActiveX والمكونات COM.
- الدعم المقدم لتطبيقات قواعد البيانات.
- الإمكانية أو الخاصية CORBA.
- إعداد الأفعال Actions التى تستجيب من خلالها التطبيقات لطلبات المستخدمين.
- المتغيرات الخاصة أو المفصلة Custom.
- وحدات وإمكانيات الـ VCL.
- وحدات وإمكانيات الـ RTL.
- إعداد تطبيقات وبرامج لديها القدرة على العمل بمختلف نظم التشغيل.
- أدوات الترجمة.
- التغييرات التى أجريت على طريقة نشر وتوزيع التطبيقات والبرامج.
- حساسية نظام المساعدة.



بعض من الإمكانيات والتحسينات تكون غير متاحة بكافة إصدارات editions لغة Delphi.

المظاهر والتحسينات الجديدة ببيئة التطوير المتكاملة IDE

لقد نالت بيئة التطوير المتكاملة IDE قدر كبير من المظاهر والإمكانيات والتحسينات الجديدة والتى تركزت فى الآتى :

- الـModules الخاصة بالبيانات.
- المشهد الشجرى أو الهرمى TreeView لخصائص وصفات الكائنات.
- محرر الكود البرمجى.
- أداة فحص الكائن.
- أدوات مشاهدة الكود البرمجى.
- مدير المشروع.
- قائمة الملف.
- صندوق حوار العناصر الجديدة New Items.
- شريط أدوات الإنترنت.
- التغييرات التى أجريت على باللية المكونات.
- الـmodules الخاصة بإعداد خريطة للمفاتيح.
- صندوق حوار خيارات بيئة العمل Environment Options.
- صندوق حوار الفهارس Directories.
- طريقة عرض محتويات القوائم.
- كتابة حزم برمجية فى مرحلة تصميم التطبيقات Design-time packages.

الـModules الخاصة بالبيانات (كافة الإصدارات)

أداة تصميم Module البيانات والتى كانت متوفرة لدى Delphi 5 تم تقسيمها إلى ثلاثة أجزاء وهى :

(١) Module البيانات القياسى (كما هو الحال فى Delphi 4) وهو يعتبر بديل لصفحة المكونات Components page.

(٢) أداة المشـاهدة الشـجرية TreeView للكـائن الـتى تعتـبر بديل للقسم الأيسر بأداة تصميم Module البيانات.

(٣) صفحة الديجرام Diagram Page التى تعتبر بديل لصفحة ديجرام البيانات.

أداة المشاهدة الشجرية TreeView للكائن (كافة الإصدارات)

هذه الأداة تقع فى الركن الأيسر العلوى لبيئة التطوير المتكاملة IDE كما هو موضح بالشكل التالى :

شكل توضيحى :

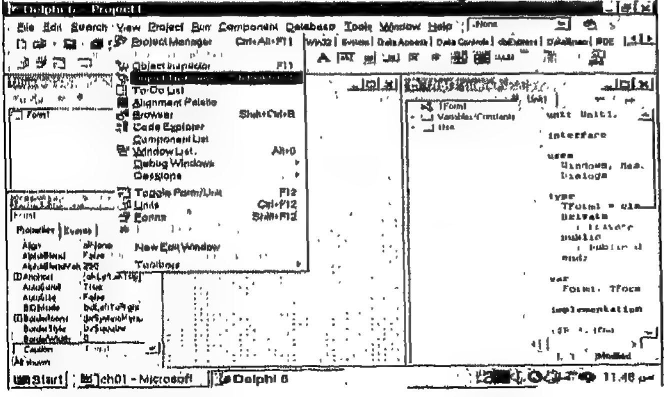
هـذا وأداة المشـاهدة الشـجرية TreeView تعتـبر ديجـرام شـجرى يقـوم بعـرض العلاقات التبادلية المنطقية بين المكونات المرئية والغير مرئية الموجودة بفورمة أو بـModule بيانات أو بإطار Frame.

تعمـل أداة المشاهدة الشجرية TreeView بطريقة متزامنة ولحظية مع أداة فحص الكـائن وأداة تصميم الفورمة ومن ثم لو أنك اخترت عنصر وقمت بتغيير المشاهدة إلى أى من الأدوات الـثلاثة سـالفة الذكـر سـتجد أن المشـاهدة تـتغير فـى نفـس الوقـت فـى الإداتـين الآخرتين. وأكثر من ذلك نقول أن صفحة الديجرام الموجودة فى محرر الكود البرمجى تعمل فقـط عـن طـريق سـحب وإسـقاط العناصـر مـن الـنافذة الخاصـة بـأداة المشـاهدة الشـجرية TreeView.

لقـد تم إضـافة المـزيد مـن المهام الوظيفية لأداة المشاهدة الشجرية TreeView ومن بين هذه المهام الوظيفية ما يلى :

- لقد تم وضع نافذة أداة المشاهدة الشجرية TreeView فوق نافذة أداة فحص الكائن Object Inspector وإذا لم تكن ظاهرة على الشاشة فى مرحلة ما فإنها تظهر فورا عندما تضغط على مجموعة المفاتيح Alt+Shift+F11 معا أو عندما تختار Object TreeView من القائمة View كما هو موضح بالشكل التالى :

شكل توضيحى :

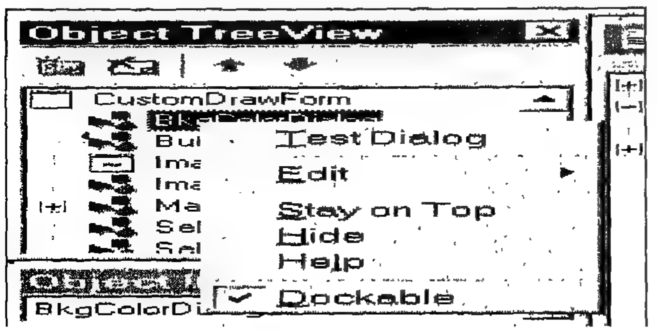


- تتضمن أداة المشاهدة الشجرية TreeView المكونات الموجودة بكل من الفورمة والإطار وModule البيانات.
- تتضمن أداة المشاهدة الشجرية TreeView المكونات المرئية والغير مرئية أيضا.
- نافذة أداة المشاهدة الشجرية TreeView تتضمن شريط أدوات جديد يشتمل على الأيقونة New Item والأيقونة Delete والأيقونة Move Up والأيقونة Move Down. وهذه الأيقونات تعمل من أجل خصائص المكون. فعلى سبيل المثال لو أنك أضفت مكون Dataset فإنك تستطيع اختيار الخاصية Aggregate ثم تنقر بالماوس على الأيقونة New Item لكى تضيف حقل جديد. ولو أن هناك أشياء مختلفة الأنواع يمكن إضافتها فى هذه الحالة سيتم فتح قائمة صغيرة عند النقر بالماوس على الأيقونة New Item.
- لقد تم إضافة المزيد من الـglyphs التى تعمل على عرض أو تمثيل نوع العنصر الذى قد يكون أب Parent (أصل) مرئى أو قد يكون مكون وليد أو تابع Child أو قد يكون مكون غير مرئى أو قد يكون خاصية ضمنية أو غير ضمنية. فعلى سبيل المثال العناصر التى يتم إنشاؤها ضمنيا بدون أن تشعر أنت مثل التعامل الافتراضى

default session تظهـر بـاللون الأبـيض والأسود فى نافذة أداة المشاهدة الشجرية TreeView.

- فـى الإصـدارات الســابقة كـنا نجـد أنه فى أثناء كون قيم خصائص أى مكون غير مكـتملة نلاحـظ أن الـــglyph الخاص بهذا المكون تكون محاطة بدائرة حمراء. أما الآن فإنـنا نلاحـظ علامـة استفهام حمـراء داخـل دائـرة صفراء تظهـر على يسار الأيقونة.
- الـنقر بالزر الأيمن للماوس على أى عنصر من العناصر الموجودة بنافذة أداة المشاهدة الشـجرية TreeView يـؤدى إلى ظهـور قائمة مختصرة تضم الخيارات الموضح فى الشكل التالى :

شكل توضيحى :



وهـذه الخـيارات نلاحـظ أنهـا أكـثر مـن التى تشتمل عليها القائمة المختصرة التى تظهـر عـند الـنقر الـزر الأيمن للماوس على أى عنصر بنافذة أداة تصميم Module البيانات كما هو موضح بالشكل التالى :

شكل توضيحى :

CustomDrawTreeView.pas
CustomDrawTreeView
TCustomDrawForm
Variables/Constants
Uses
unit CustomDra
New
Rename
View Editor Shift+Ctrl+E
Dockable
Properties

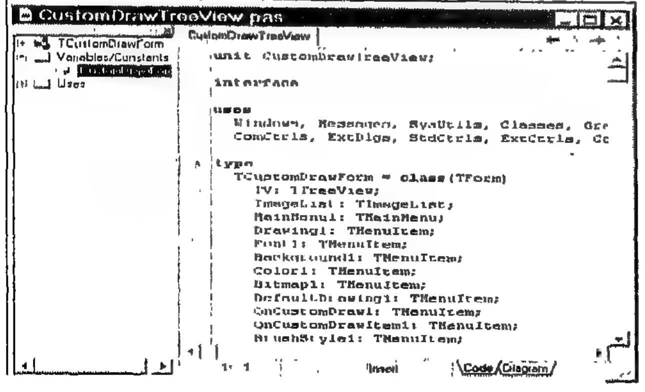
محرر الكود البرمجي

فيما يلي سوف نستعرض سويا المظاهر الجديدة التي أضيفت لمحرر الكود البرمجي من خلال Delphi 6.

أدوات تصميم السطح (الإصدار الفني Professional والإصدار المتكامل Enterprise)

محـرر الكود البرمجي، Code Editor أصبح الآن يقدم الدعم الكامل لأدوات تصميم السـطح Surface designers أو للحزمة المحملة للمشاهد الخاصة أو المفصلة. وهذه المشاهد يـتم الوصـول إلـيها مـن خـلال مجموعـة مـن التبويبات الموجودة في سسطر الحالة كما هو موضح بالشكل التالي :

شكل توضيحي :

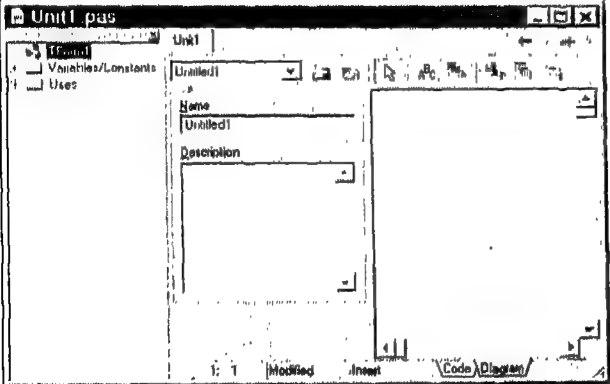


المشـهد الأساسـي، built-in الوحـيد هـو صفحة الكود البرمجي، القياسية. ونود هنا القـول بأنـه بـناء عـلى، الاصـدار الـذى تـتعامل معه من لغة Delphi فإنك تستطيع الوصول للصفحات التالية بمحرر الكود البرمجي :

صفحة الديجرام (الإصدار الفني Professional والإصدار المتكامل Enterprise)

الشـكل التالي يوضح لنا صفحة الديجرام الموجودة في محرر الكود البرمجي Code Editor

شكل توضيحي :





وهـذه الصفحة تعمل على توفير العديد من الأدوات المرئية Visual من أجل إعداد وتهيـئة ديجرام من المستطيلات والمربعات والخطوط لعرض العلاقات التبادلية بين المكونات المرئـية والمكونـات غـير المرئية. ويمكن اعتبار صفحة الديجرام بمثابة أداة توثيق حيث إنها تقـوم بتوضـيح هـذه العلاقـات التبادلـية تخطيطيا Schematically كما إنها تسمح لك بأن تضـيف أى تعلـيقات ترغـبها للديجـرام. ونـود هـنا القول بأن المكونات لا تظهر فى صفحة الديجرام حتى تقوم برسمهم من خلال أداة المشاهدة الشجرية TreeView للكائن.

المظاهر والإمكانيات الجديدة لدى صفحة الديجرام :

- تسـتطيع أن تختار العديد من العناصر فى نفس الوقت من نافذة المشاهدة الشجرية للكائن Object TreeView ثم تسحب هذه العناصر إلى داخل صفحة الديجرام.
- الجانـب الأيسـر بصفحة الديجرام أصبح مشتملا على حقل تستطيع فيه أن تكتب اسـم ووصـف لكـل ديجـرام تقـوم بإنشاؤه كما إنه يشتمل أيضا على قائمة منسدلة للعثور على الديجرامات التى تم إنشاؤها وتسميتها من قبل.
- لقـد تم إضـافة شـريط الأدوات يضم العديد من الأيقونات إلى صفحة الديجرام. وهذه الأيقونات لعمل الوصلات بين العلاقات التبادلية ولتحديد مواضع بلوكات تعليق داخل الديجرام.
- يتم تلقائيا تخصيص عناوين لأدوات وصل الصفات Property connectors. وأنت تستطيع سحب العناوين وتسقطها فى أى موضع بصفحة الديجرا م.
- تسـتطيع أن تنشأ ديجرام لكل Module للبيانات أو لكل فورمة أو لكل إطار قمت بإضافته إلى المشروع.

صفحات الـWebSnap (الإصدار المتكامل Enterprise)

عـند إنشـاء تطبـيق مـن أجـل أن يعمـل فـى أحـد خـوادم الويـب باسـتخدام الـ WebSnap فـإن المكونـات الـتى توجـد فـى module صـفحة الويـب تعمـل عـلى تكويـن صفحات لكل من الآتى :

- الـScript المعد بلغة HTML.
- ناتج تنفيذ الكود البرمجى المكتوب بلغة HTML.

- المعاينة Preview.
- التسلسل الشجرى XML Tree.
- التسلسل الشجرى XSL Tree.

تبويبات السحب والإسقاط (كافة الإصدارات)

تبويبات السحب والإسقاط عبارة عن وحدة التبويبات Tabs الموجودة فى الجانب العلوى بمحرر الكود البرمجي والتى يمكن الشعور بها من خلال سحب وإسقاط هذه التبويبات. فعلى سبيل المثال لو أن لديك وحدتين إحداهما تسمى About والأخرى تسمى TextEditor فى هذه الحالة تستطيع سحب الوحدة About إلى الجانب الأيمن للوحدة TextEditor.

أداة فحص الكائن Object Inspector (كافة الإصدارات)

نافذة أداة فحص الكائن Object Inspector أصبحت الآن موجودة أسفل نافذة المشاهدة الشجرية للكائن Object Tree وهى موضحة فى الشكل التالى :

شكل توضيحى :

Object Inspector

Form1

Properties | Events

Property	Value
Align	alNone
AlphaBlend	False
AlphaBlendValu	255
⊞ Anchors	[akLeft,akTop]
AutoScroll	True
AutoSize	False
BiDiMode	bdLeftToRight
⊞ BorderIcons	[biSystemMenu,
BorderStyle	bsSizeable
BorderWidth	0
Caption	Form1

All shown

فيما يلى سوف نستعرض سويا المظاهر والإمكانيات الجديدة التى أضيفت لأداة فحص الكائن Object Inspector من خلال Delphi 6.

قائمة عرض اللحظية Instance list box

لقد تم إضافة قائمة منسدلة لحظية بالجزء العلوى لنافذة أداة فحص الكائن Object Inspector وهى تتمتع بالمظاهر والإمكانيات التالية :

- تعمـل القائمـة اللحظـية عـلى استعراض اسم القطاع لكل كائن موجودة فى القائمة وليس فقط للكائن الموجود فى قمة القائمة.
- تسـتطيع أن تمـنح أى مكـون نفـس الأسـم الممنوح لأى فورمة أو Module بيانات موجود به. فعلى سبيل المثال تستطيع أن تضيف مكون أيقونة إلى الفورمة Form1 ثـم تقوم بتغيير اسمه ليصبح Form1 أيضا وفى هذه الحالة كلا الأسمين سيظهران فى القائمة اللحظية.
- تقـوم القائمـة اللحظـية بعـرض أداة تعلـيق ToolTip للمكون المختار حاليا ويعتبر ذلك مفيدا فى حالة أن اسم المكون كان أعرض من القائمة اللحظية.
- يمكن إخفاء القائمة اللحظية.

صندوق حوار الخصائص

قائمـة المحـتوى Context الخاصـة بنافذة أداة فحص الكائن Object Inspector أصبح لديهـا صـندوق حـوار خصائص جديد والذى يمكن الوصول إليه أيضا عن طريق فتح القائمـة Tools واختـيار مـنها الأمـر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :

شكل توضيحى :

Environment Options

Type Library | Environment Variables | Delphi Direct | Internet
Preferences | Designer | Object Inspector | Palette | Library | Explorer

Autosave options
Editor files
Project desktop

Desktop contents
Desktop only
Desktop and symbols

Docking
Auto drag docking

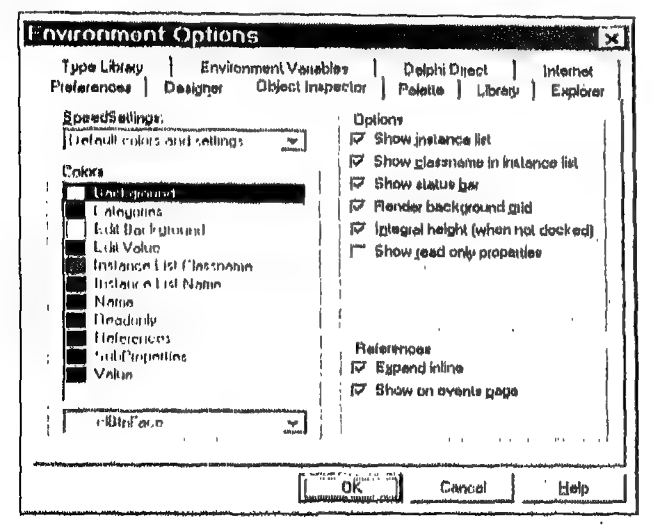
Compiling and running
Show compiler progress
Warn on package rebuild
Minimize on run
Hide designers on run

Shared repository
Directory: Browse...

OK Cancel Help

وفـيه انقـر بالمـاوس عـلى التبويب Object Inspector ليظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :



وهذا التبويب يضم العديد من خيارات العرض التالية :

- مجموعـة الــSpeedSettings التى تعمل على تفصيل الألوان الخاصة بأداة فحص الكائن Object Inspector.
- عـرض أو إخفـاء كـل مـن القائمـة اللحظية وأسماء القطاعات فى القائمة اللحظية وشريط الحالة وشبكة نقط الخلفية والخصائص التى تكون للقراءة فقط.
- خصـائص مرجعيات الكـائن يمكـن تمديدهـا inline وعرضـها فى كل من صفحة الخصائص وصفحة الأحداث Events.

مرجعيات المكون الممتدة Inline

تعمـل مرجعـيات المكـون المـمتدة Inline على عرض كل من الخصائص والأحداث الخاصـة بـأى كـان مرجعى وذلك بدون الحاجة للقيام فعليا باختيار المكون المرجعى نفسه. وفى هذا الجزء نجد المظاهر والإمكانيات الجديدة التالية :

- الخصـائص الـتى تشير مرجعـيا لكـائن درجـة ثانـية تكـون حمـراء اللـون كما أن الخصائص الخاصة بهذا الكائن الدرجة الثانية تكون بطبيعة الحال خضراء اللون.



- الأحداث التى تشير مرجعيا لكائن درجة ثانية تكون حمراء اللون كما أن الأحداث الخاصة بهذا الكائن الدرجة الثانية تكون بطبيعة الحال خضراء اللون.
- الخصائص التى تتفاعل مع بعضها البعض يمكن الإشارة المرجعية إليها inline.

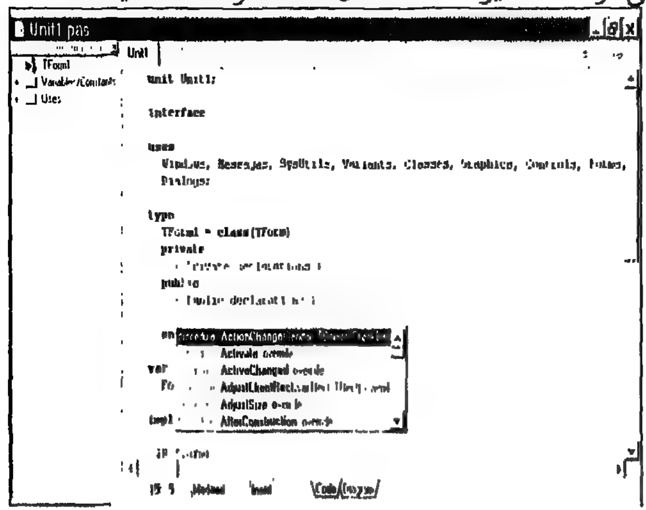
أدوات التعامل مع الكود البرمجى Code insight tools (كافة الإصدارات)

إمكانية التعامل مع الكود البرمجى أصبحت لديها إمكانيات ومظاهر تم تحديثها وخاصة بالنسبة لتكملة الكود ومعاملات الكود code parameters.

تكملة الكود البرمجى Code completion

الآن عندما تقوم بنفسك باستدعاء خاصية تكملة الكود البرمجى (بالضغط على المفتاحين Ctrl+Space) بلوحة المفاتيح وأنت لاتزال فى سطر فارغ داخل جسم أى روتين فستجد أنه تظهر على الفور قائمة منسدلة (الموضح فى الشكل التالى) تقدم كل رموز من وحدات الـRTL الإضافية حتى لو كانت غير مستخدمه من خلال الوحدة الحالية :

شكل توضيحى :



فيما يلى سنستعرض سويا المظاهر والإمكانيات الجديدة بهذه الخاصية :

- تصفية وفرز كافة الإعلانات عن طرق التفاعل interface التى يتم الإشارة إليها مرجعيا من خلال قطاعات صفة القراءة أو الكتابة. فالقائمة المنسدلة تعرض فقط الخصائص والطرق المستقلة stand-alone التى تم الإعلان عنها فى نوعية التفاعل interface type.

- إجـراء عملـية تصـفية وفـرز فى أثناء الكتابة بمحرر الكود البرمجى. كما أن النص يتم إزالته القائيا عندما تختار أى عنصر من القائمة المنسدلة.
- العمـل داخـل مـنطقة تعـريف القطـاع Class فـى قسـم الـتفاعل مـع تقديم الدعم لإمكانية الاختيار الكتعدد multiselect.
- عرض الطرق المتوارثة والتصورية بالإضافة إلى طرق التفاعل والخصائص.
- إمكانية تغيير حجم القائمة المنسدلة.
- لقـد تم إضـافة الألـوان للمساعدة فى التمييز بين مختلف العناصر الموجودة بالقائمة المنسـدلة. فعـلى سبيل المثال الإجراءات تكون ملونة باللون الأزرق المخضر teal فى حـين أن الـدوال تكـون ملونـة بـالأزرق الغامق. ولكن على العموم تستطيع أن تقوم بتغيير الألوان الإفتراضية فى أحقية الاستخدام Registry.
- تقديـم الدعـم لأسـاليب الرسائل WM_xxx وCM_xxx وCN_xxx المعتمدة على الثوابت المتشابهة الأسماء والموجودة بكافة الوحدات بالجملة Uses.
- أسـاليب الاسـتخلاص أصبحت الآن تظهر باللون الأحمر فى مناطق الإعلانات عن الأنواع Type declarations. أما باقى كافة الـcontexts الأخرى الخاصة بأدوات الـتعامل مـع الكـود الـبرمجى تعمـل عـلى إظهـار أساليب الاستخلاص باستخدام الألوان المعتادة لكل من الإجراءات والدوال.

معاملات الكود البرمجى code parameters

- لو أن عنصر كان عبارة عن إجراء أو دالة يستخدم عدد من المعاملات parameters فـي هـذه الحالة يتم إضافة "(" إلى اسم هذا العنصر كما يتم عرض على الفور أداة تعليق مشتملة على الكود البرمجى للمعامل.

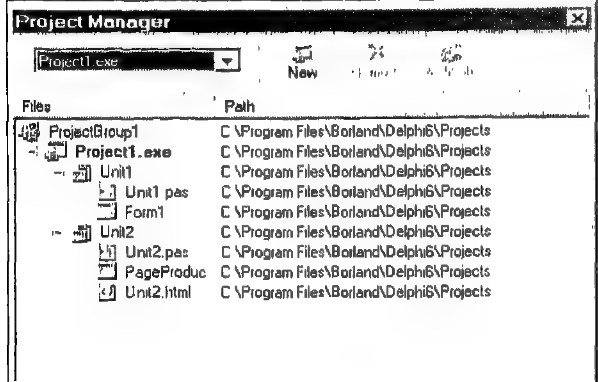
الـmodule الجديد لتخطيط المفاتيح Key mapping (كافة الإصدارات)

لقـد تم إضافة مجموعة تخطيط المفاتيح الخاصة بلغة VB (Visual Basic) إلى الـ modules الخاصة بتخطيط المفاتيح الموجودة بالفعل لدى لغة Delphi.

المزم Packages فى مدير المشروع (كافة الإصدارات)

الآن نجد أن كافة حزم المشاريع المفتوحة حاليا تكون معروضة فى مشهد مدير المشروع كما هو موضح بالشكل التالى :

شكل توضيحى :



ونود هنا القول بأن هذه الـnodes التى تشير مرجعيا إلى حزم المشروعات يمكن استخدامها للمساعدة فى متابعة وإدارة المشروع النشط حاليا كما يمكن الاستعانه بها أيضا كأداة للتجول لكى تصل إلى أى حزمة مفتوحة حتى عندما تكون هذه الحزمة ليست عضوا فى مجموعة المشروع الحالى.

قائمة الملف File Menu

لقد تم تحديث قائمة الملف File Menu وهذا التحديث يتمثل فى إضافة القائمة الفرعية New للقائمة File كما هو موضح بالشكل التالى :

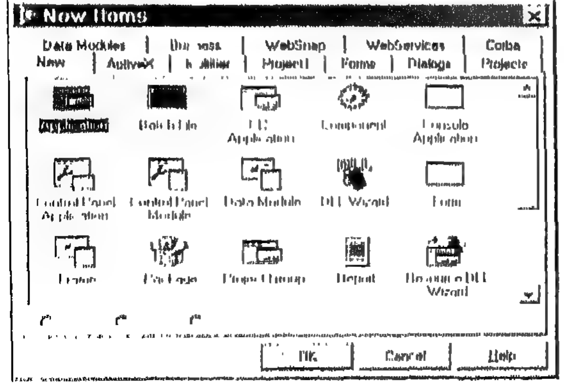
شكل توضيحى :

وكما نـرى فـإن هـذه القائمـة الفرعـية تشـتمل عـلى ٦ أنواع من المشاريع (تطبيق Application وتطبـيق CLX وModule بـيانات وفورمـة وإطار ووحدة) بالإضافة إلى باقى الاختيار Other الذى يعمل على فتح صندوق حوار العناصر الجديدة New Items.

صندوق حوار العناصر الجديدة New Items

الشكل التالى يوضح لنا صندوق حوار العناصر الجديدة New Items أو الذى يعرف بمخزن أو مستودع الكائنات Object Repository :

شكل توضيحى :

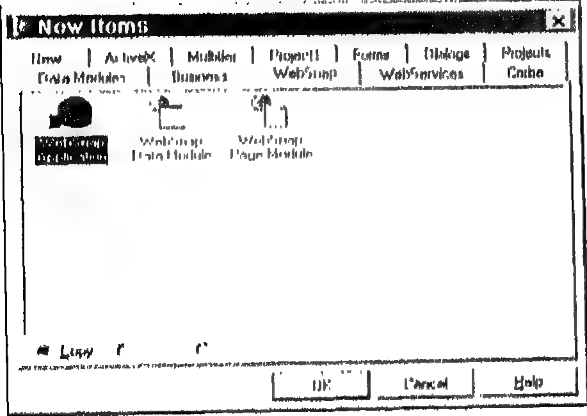


كما نرى أن صندوق الحوار New Items أصبح مشتملا على ثلاثة تبويبات جديدة بالإضـافة إلى عـدد من المعالجات Wizards الجديدة وعدد من modules البيانات الجديدة التالية :

التبويب WebSnap :

الشكل التالى يوضح لنا محتويات هذا التبويب الجديد:

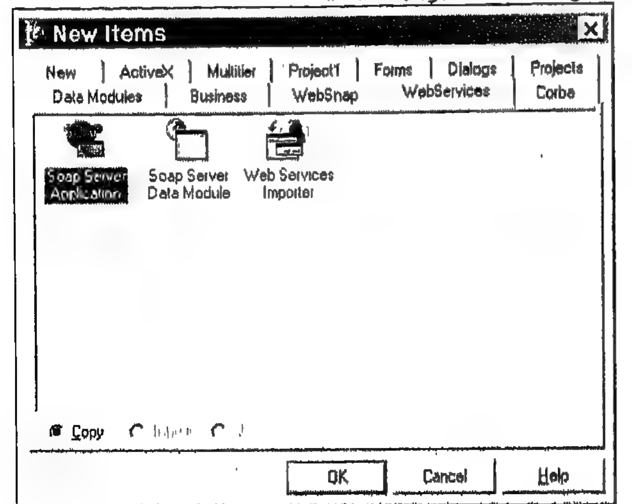
شكل توضيحى :



التبويب Web Services :

الشكل التالى يوضح لنا محتويات هذا التبويب الجديد:

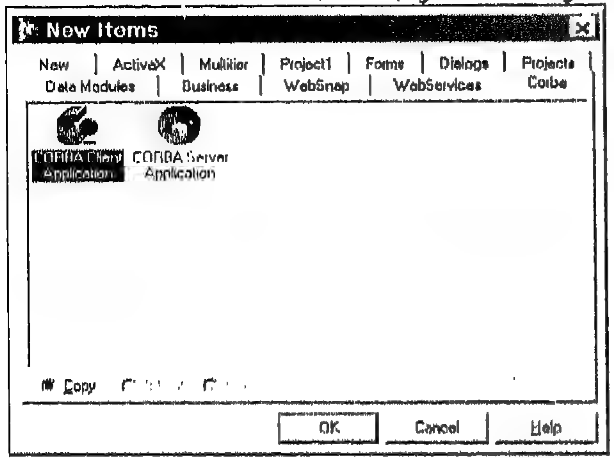
شكل توضيحى :



التبويب CORBA :

الشكل التالى يوضح لنا محتويات هذا التبويب الجديد:

شكل توضيحى :





تستطيع أيضا أن تصل لهذه المعالجات من خلال الأمر regenerate CORBA IDL Files الموجود بالقائمة Tools والذى يؤدى لفتح صندوق الحوار Regenerate CORBA IDL Files الموضح فى الشكل التالى :

شريط أدوات الإنترنت (الإصدار المتكامل Enterprise)

شـريط أدوات الإنترنـت (الموضـح فـى الشكل التالى) يعتبر شريط أدوات جديد وهو يساعدك فى إنشاء تطبيق WebSnap للعمل بخوادم الويب :

شكل توضيحى :



إذا كان شريط أدوات الإنترنت غير ظاهر على الشاشة فى هذه الحالة افتح القائمة View ومنها اختر Toolbars ثم علم بالماوس على الاختيار Internet.

الأيقونـات الموجـودة بشـريط أدوات الإنترنـت تجعلـك تصـل إلى ثلاثـة معالجـات جديدة وهى :

- المعالج New WebSnap App.
- المعالج New WebSnap Page Module.
- والمعالج New WebSnap Data Module.
- المحرر الخارجى External Editor.



هذه الأيقونات تناظر الدوال المستخدمة دوما لإنشاء تطبيقات خوادم الويب من خلال الـWebSnap.

التغييرات التى أجريت على باللينة المكونات

- الصفحة الإضافية Additional Page أصبحت تمتلك الآن مكونات action band جديدة وذلك لتكوين قوائم الأفعال ومعالجتها وكذلك لبناء قوائم وشرائط أدوات مفصلة أو خاصة بما فيها كل من TActionManager و TActionMainMenuBar وTActionToolBar وTCustomizedDlg.
- بالنسبة للتطبيقات CLX يتم استخدام صفحة أدوات التحكم الشائعة الاستخدام Common Controls Page بدلا من الصفحة Win32 (الإصدار الفنى Professional والإصدار المتكامل Enterprise).
- تم إضافة الصفحة WebSnap الجديدة والتى تشتمل على مكونات يمكن استخدامها لبناء تطبيقات خوادم الويب مع العديد من الـModules (الإصدار المتكامل Enterprise فقط).
- تم إضافة صفحة خدمات الويب WebServices Page لكى يتمكن المبرمج من كتابة تطبيقات متعددة الطبقات Multitier للعمل من خلال شبكة من الحاسبات الآلية أو من خلال شبكة الويب (الإصدار المتكامل Enterprise فقط).
- العديد من التغييرات تم إجراؤها على صفحات باللينة مكون قاعدة البيانات database Component palette pages (الإصدار الفنى Professional والإصدار المتكامل Enterprise).
- تم إضافة الصفحة COM + Page الجديدة للمكون المسمى COMAdminCatalog (الإصدار الفنى Professional والإصدار المتكامل Enterprise).
- تم إضافة ثلاث صفحات جديدة لبروتوكولات الاتصال عبر الإنترنت وهى الصفحة Indy Clients والصفحة Indy Servers والصفحة Indy Misc. ونود هنا القول بأن مكونات الإنترنت المباشرة (Indy) ما هى إلا بروتوكولات لفتح المصادر الإنترنت المتعددة والقائمة على أساس التكنولوجيا Blocking Sockets (الإصدار الفنى Professional والإصدار المتكامل Enterprise).

صندوق حوار خيارات بيئة العمل Environment Options

الشكل التالى يوضح صندوق حوار خيارات بيئة العمل Environment Options (الذى يتم الوصول إليه من خلال الأمر Environment Options بالقائمة Tools) :

شكل توضيحى :

Environment Options

Type Library | Environment Variables | Delphi Direct | Internet
Preferences | Designer | Object Inspector | Palette | Library | Explorer

Autosave options
Editor files
Project desktop

Desktop contents
Desktop only
Desktop and symbols

Docking
Auto drag docking

Shared repository
Directory: Browse...

Compiling and running
Show compiler progress
Warn on package rebuild
Minimize on run
Hide designers on run

OK | Cancel | Help

صندوق الحوار هذا أصبح الآن مشتملا على مجموعة التبويبات جديدة التالية :

تبويب المصمم Designer :

الشكل التالى يقدم لنا محتويات هذا التبويب :

شكل توضيحى :

Environment Options

Type Library | Environment Variables | Delphi Direct | Internet
Preferences | Designer | Object Inspector | Palette | Library | Explorer

Grid options
Display grid
Snap to grid
Grid size
X: 8 Y: 8

Module creation options
New forms as text
Auto create forms & data modules

Options
Show component captions
Show designer hints
Show extended control hints

OK | Cancel | Help

نلاحظ أنه قد تم نقل الخيارات الخاصة بمصمم الفورمة من التبويب Preferences إلى هذا التبويب كما نلاحظ أيضا وجود اختيار جديد وهو الاختيار Show extended control hints.



تبويب متغيرات بيئة العمل Environment Variables :

الشكل التالى يقدم لنا محتويات هذا التبويب :

شكل توضيحى :

فى هذا التبويب تستطيع أن تشاهد المتغيرات الخاصة ببيئة العمل الخاصة باللغة كما تستطيع أيضا إنشاء وتحرير ومسح التعديات التى قد يقوم بها المستخدم على متغيرات النظام.

تبويب أداة فحص الكائن Object Inspector:

الشكل التالى يقدم لنا محتويات هذا التبويب :

شكل توضيحى :

من خلال هذا التبويب تستطيع تحديد القيم المفضلة Preferences الخاصة بأداة فحص الكائن وذلك باستخدام صندوق الحوار Properties الجديد والذى يمكن الوصول إليه أيضا عن طريق النقر بالزر الأيمن للماوس على نافذة أداة فحص الكائن ثم اختيار Properties من القائمة المختصرة التى تظهر على الشاشة.

تبويب الإنترنت Internet:

الشكل التالى يقدم لنا محتويات هذا التبويب :

شكل توضيحى :

تستطيع من خلال هذا التبويب تحديد الأفضليات Preferences الخاصة بأنواع الملفات وخيارات الـ Script التى ترغب فى استخدامها من أجل تطبيقات الـ WebSnap التى تتولى إعدادها.

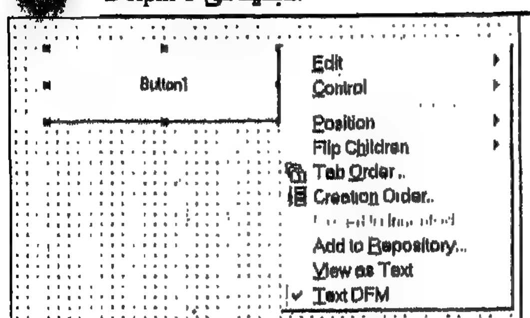
صندوق حوار الفهارس Directories

صندوق حوار الفهارس Directories الخاص بتحرير العناصر أصبح الآن مثله مثل مكتبة المسار Library Path يقم بعرض المسارات الغير صحيحة بلون رمادى (تكون غير نشطة). وأنت تستطيع إزالة المسارات الغير صحيحة من خلال المفتاح Delete Invalid Paths.

عرض قائمة المحتوى Context menu

فى أثناء كون كائن مختار داخل فورمة أو فى حالة كون نافذة مختارة أو فى حالة كون دفة التحكم متركزة على أى عنصر فى بيئة التطوير المتكاملة IDE تستطيع الآن أن تعرض قوائم المحتوى الخاصة بالكائن أو العنصر المختار وذلك عن طريق الضغط على المفتاحين Alt+F10 بلوحة المفاتيح أو بالضغط على المفتاح Menu بلوحات مفاتيح مايكروسوفت. هذا وفيما يلى الشكل التالى يوضح لنا واحدة من قوائم المحتوى التى تظهر على الشاشة :

شكل توضيحى :



كتابة الحزم البرمجية فى أثناء مرحلة التصميم Design-time packages

ينبغى عليك إضافة الملف DesignIDE.dcp إلى قائمة المتطلبات الخاصة بالحزمة البرمجية التى تود إعدادها.

مظاهر وإمكانيات الإنترنت الجديدة (الإصدار الفنى Professional والإصدار المتكامل Enterprise)

فيما يلى سنتعرف سويا على مظاهر وإمكانيات الإنترنت الجديدة المضافة حديثا للغة Delphi 6.

الدعم الخاص بخدمات الويب (الإصدار المتكامل Enterprise فقط)

الآن يمكن القول بأن المكونات والتغييرات التى أجريت على المترجم الخاص بلغة Delphi 6 تجعلنا قادرين على كتابة التطبيقات التى تعرف بأنها خدمات الويب كما تسمح لك بأن تقدم الدعم لكتابة تطبيقات الـclients التى تصل إلى خدمات الويب باستخدام SOAP.



خدمات الويب تعتبر تكنولوجيا جديدة ناشئة. وبسبب هذا فإن المكونات التى تعمل على تدعيم هذه الإمكانية أو التكنولوجيا بما فيها الوسائط Interfaces تكون معرضة للتغيير فى أى وقت وباستمرار.

الدعم الخاص بتطبيقات خادم الويب (الإصدار الفنى Professional والإصدار المتكامل Enterprise)

يتضمن Delphi 6 مضيف لمظاهر الإنترنت الجديدة من أجل بناء تطبيقات خادم الويب بما فيها التغييرات التى تجرى على المظاهر والإمكانيات WebBroker الموجودة بالفعل وأى مجموعة جديدة من المظاهر والإمكانيات التى يطلق عليها WebSnap.

خادم الويب Apache

(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

كل من الـWebSnap والـWebBroker أصبحا الآن يقدمان الدعم لخادم الويب Apache بالإضافة إلى تقديم الدعم لكل من ISAPI وCGI وWinCGI.

أداة معالجة الثغرات لتطبيقات الويب Web Application Debugger

(الإصدار الفنى والإصدار المتكامل)

أداة معالجة الثغرات لتطبيقات الويب Web Application Debugger تمكنك من مراقبة الطلبات HTTP والاستجابات بالإضافة إلى الفترة الزمنية للاستجابة كما إنها تمكنك أيضا من تطوير وإعداد تطبيقات من خلال كل من الـWebSnap والـWebBroker وذلك بدون إقامة خادم ويب تجارى. ونود هنا القول بأن عملية معالجة الثغرات Debugging بأى تطبيق من تطبيقات خادم الويب يمكن أن تكون صعبة إلى حد ما وذلك بسبب أن التطبيق الذى تتولى إعداده يحتاج لأن يعمل من خلال استجابه لرسالة صادرة من أحد خوادم الويب. ومن ثم فإن أداة معالجة الثغرات لتطبيقات الويب تعمل على محاكاة مثل هذه الرسالة.

المظاهر والإمكانيات WebBroker المحسنة

(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

الـWebBroker أصبحت الآن :

- تعمل على تمكينك من كتابة تطبيقات خوادم الويب من أجل التطبيقات CLX.
- لديها القدرة على التوافق مع Modules الويب بكل من الإصدار الثالث والرابع والخامس من لغة Delphi مع وجود بعض القيود. فعلى سبيل المثال المشاريع القديمة قد تحتاج لإضافة مستخدمين جدد أو لتغيير مستخدمين حاليين. ونود هنا القول بأن Modules الويب المتعددة لن يتم تدعيمها بالتطبيقات المعدة بالإصدارات السابقة للغة Delphi.

مظاهر الـWebSnap الجديدة (الإصدار المتكامل Enterprise فقط)

الـWebSnap تكون معتمدة على مظاهر وإمكانيات الـWebBroker لدى الإصدار الخامس للغة Delphi بالإضافة إلى الكثير من المظاهر والإمكانيات الوظيفية الجديدة والتى تتضمن الآتى :



- الـModules المتعددة.
- المعالجات الجديدة.
- إعداد Scripts للعمل فى خوادم الويب.
- المكونات الجديدة.
- أدوات تصميم السطح.

Modules الويب المتعددة

تطبيق واحد من تطبيقات المعدة من خلال الـWebSnap تكون لدية القدرة على تدعيم العديد من Modules الويب والتى تعمل على تقسيم التطبيق إلى وحدات كما إنها تسمح أيضا للعديد من المبرمجين بأن يعملوا بنفس المشروع مع التقليل بقدر الإمكان من التداخلات الغير مقصودة Conflicts. وأكثر من ذلك ما يلى:

- إمكانية تدعيم الحالات المتعددة Multiple Instance للطلبات المتلاحقة فى فترة زمنية صغيرة إلى حد ما.
- تدعيم الإشارات المرجعية للكائنات بين الـModules.
- يتم إنشاء الـModules تلقائيا وذلك بهدف خدمة طلب أو عند الإشارة المرجعية لهم من خلال Module آخر.

المعالجات الجديدة لـModule الويب

من خلال معالج تطبيق خادم الويب تستطيع أن تقوم ببناء تطبيق مع Module ويب مفصل أو مع Module تطبيق ويب أو مع Module صفحة الويب وذلك باستخدام المكونات الأربعة التالية :

- المكون TWebAppPageModule.
- المكون TwebAppDataModule.
- المكون TWebPageModule.
- المكون TWebDataModule.

بالنسبة لـModule بيانات الويب فهو يقوم بإنشاء حاوية Container للمكونات التى تكون متاحة للاستخدام المشترك عبر تطبيق الويب الذى تتولى إعداده. ولكن هذا الـModule ليس لدية ملف قالب HTML ملحق به.

أما Module صفحة الويب فيعمل على تعريف صفحة ويب جديدة بالتطبيق الذى تـتولى إعـداده. وهـذا الـModule لدية ملف قالب HTML ملحق به كما يعرض الـScript الذى تتولى كتابته باستخدام أدوات تصميم سطح الـWebSnap. كذلك فإن هذا الـModule يعمـل أيضا عـلى تعريف اسم أى صفحة من الصفحات المنطقية وهو لا يحتاج لتعريف أى فعـل action للوصـول لصـفحة الويب كما إنه يعمل أيضا على إدارة ملف القالب HTML من خلال نظام تصورى للملفات وعرضه بنافذة مدير المشروع.

هـناك أكـثر مـن طـريقة لكى تصل لمجموعة معالجات Module الويب ومنها فتح القائمة File ثم اختيار New ثم Other وفى صندوق الحوار New Items انقر بالماوس على التبويـب WebSnap ثـم الـنقر المزدوج بالماوس على الأيقونة WebSnap Application ليظهر على الشاشة صندوق الحوار New WebSnap Application الموضح فى الشكل التالى :

شكل توضيحى :

New WebSnap Application

Server Type
- ISAPI/NSAPI Dynamic Link Library
- CGI Stand-alone executable
- Win-CGI Stand-alone executable
- Apache Shared Module (DLL)
- Web App Debugger executable

Application Module Components
- Page Module
- Data Module

Components...

Application Module Options

Page Name: PageProducerPage2

Page Options...

Caching: Cache Instance

Default | OK | Cancel | Help

هـناك طـريقة أخـرى للقيام بـنفس المهمـة وهـى فـتح القائمـة View ثـم اخـتيار Toolbars ثـم التعليم بالماوس على الاختيار Internet ثم النقر بالماوس على الأيقونة New WebSnap Application



إعداد Scripts للعمل بخوادم الويب

تتضمن الـWebSnap إداة إعداد Scripts للعمل بخوادم الويب وهى عبارة عن لغة بسيطة لإعداد الـScripts تم تصميمها خصيصا من أجل برمجة صفحات الويب. وهى تتميز بالآتى :

- إمكانية إعداد Scripts تكون لديها القدرة على الوصول للمكونات الخاصة بلغة Delphi.
- تقديم الدعم اللازم لمولد الـScripting الفعال كما هو الحال بالنسبة للغة VB أو لغة JavaScript.
- إمكانية تدعيم الـScript بلغة HTML.

المكونات الجديدة

تأتى الـWebSnap مع مضيف لعدد من المكونات الجديدة والتى تتضمن أداة تحفيز الأفعال Dispatcher والمنظم adapter وأداة إنتاج الصفحة والـSession وقائمة المستخدم.

مكونات أداة تحفيز الأفعال Dispatcher

مكونات أداة تحفيز الأفعال Dispatcher تعالج تلقائيا أنواع مختلفة من الطلبات الخاصة بمحتوى سفحة الويب والمهام الفرعية للفورمة والتى تتم من خلال لغة HTML بالإضافة إلى الطلبات الخاصة بالصور الديناميكية.

المكونات الجديدة والخاصة بأداة تحفيز الأفعال Dispatcher تتضمن الآتى :

- المكون TPageDispatcher.
- المكون TAdapterDispatcher.
- المكون TWebDispatcher. وهذا المكون يمتلك الحدث الجديد OnException والتى يتم التعامل معه بالصورة التالية TCustomWebDispatcher.OnException.

مكونات المنظم Adapter Components

المنظمات Adapters تعمل على توفير وسائل لتعريف وسيط أو واجهة استخدام Interface تكون قابلة للبرمجة على حسب قوانين وقواعد العمل وذلك من أجل التطبيق

الـذى تـتولى إعـداده. فعـلى سبيل المثال يمكن استخدام المكون TDataSetAdapter لإعداد مكونات Dataset تكون قابلة للبرمجة من خلال لغات إعداد الـScripts.

المكونات الجديدة الخاصة بالمنظم تتضمن الآتى :

- المكون TAdapter.
- المكون TApplicationAdapter.
- المكون TPagedAdapter.
- المكون TLoginFormAdapter.
- المكون TEndUserAdapter.
- المكون TEndUserSessionAdapter.
- المكون TStringsValueList.
- المكون TDataSetValueList.

مكونات أداة إنتاج صفحة الويب Page Producer Components

أدوات بـناء الصـفحات تعمـل عـلى بناء فورم معقدة وجداول للتعامل مع البيانات كما إنها تستخدم XSL لتكوين صفحة ويب. وهذه المكونات الجديدة تتضمن الآتى :

- المكون TAdapterPageProducer.
- المكون TXSLPageProducer.

مكونات الـSession

من خلال مكونات الـSession يمكن متابعة المستخدمين النهائيين التطبيقات التى تتولى إعدادها. والمكونات الجديدة تتضمن الآتى :

- المكون TSessionsService.

مكونات قائمة المستخدم User List

تعمـل مكونـات قائمـة المسـتخدم User List عـلى توفـير طريقة للوصـول لأسمـاء المستخدمين وكلمات السر الخاص بهم وحقوق الاستخدام أو الوصول الممنوحة لهم. ونود هنا القول بأن المكونات الجديدة فى هذا الصدد عبارة عن الآتى :

- المكون TWebUserList.

أدوات تصميم السطح بـWebSnap

عندما تقوم بإعداد Script لتطبيق معد لكى يعمل بأحد خوادم الويب وذلك من خلال الـWebSnap فإنك تستطيع بناء صفحات ويب ومشاهدة النتائج فى مرحلة التصميم Design-Time داخل محرر الكود البرمجى Code Editor وبصفحات تحرير الكود البرمجى التالية :

- HTML Script.
- HTML Result.
- Preview
- XML Tree
- XSL Tree

الدعم الخاص بلغة XML (الإصدار المتكامل Enterprise فقط)

معالج ربط بيانات الـXML

هذا المعالج الجديد يقوم بإنشاء كود مكتوب بلغة Delphi وهذا الكود يعمل على توفير طريقة بسيطة للغاية للوصول لملفات البيانات المعدة بلغة XML والعمل على تحديثها أيضا. وفى هذا الصدد نقول أن الكود البرمجى الذى يتم تكوينه يمثل خاصية طبيعية وبديهى وقائمة على نموذج برمجى يجعل التعامل مع البيانات المعدة بلغة XML كما لو كنا نقوم بالبرمجة من خلال أحد مكونات لغة Delphi.

أدوات ربط البيانات المعدة بلغة XML بالمكونات الخاصة بلغة Delphi تعد بديل للتكويد البرمجى المعقد والذى يكون مطلوبا وضروريا لاستخدام واجهات الاستخدام أو الوسائط Interfaces الخاصة بنموذج الكائن لدى لغة XML (المعروف بـDOM والذى يعد اختصارا لـDocument Object Model) كما أن هذه الأدوات تعتبر مناسبة جدا للتطبيقات التى تتولى مهمة معالجة المستندات المعدة بلغة XML.

برمجة المستندات المعدة بلغة XML

أدوات ربط البيانات المعدة بلغة XML بالمكونات الخاصة بلغة Delphi تكون معتمدة على المكون الجديد المسمى TXMLDocument وكافة الوسائط أو واجهات الاستخدام الملحقة به وهى XMLDocument وXMLNode. وهذا المكون يعمل على توفير نسخة مبسطة من واجهات الاستخدام أو الوسائط الخاصة بالنموذج DOM الذى أشرنا إليه سلفا وذلك من أجل التعامل مع البيانات المعدة بلغة XML.

كل ما يطلب منك فقط هو أن تقوم بإسقاط (وضع) الكائن TXMLDocument فى الفورمة التى تتعامل معها ثم تقوم بتحديد قيمة الخاصية FileName وتجعل الخاصية Active عبارة عن True وفى النهاية سيكون لديك حالة وصول لكافة البيانات التى تتولى إعدادها من خلال لغة XML.

برمجة الكائن DOM للعمل عبر أنظمة التشغيل المختلفة

عند أدنى مستوى فى الدعم الخاص ببرمجة المستند المعد بلغة XML نجد وحدة تفاعل جديدة يطلق عليها xmldom.pas وهذه الوحدة تعمل على توفير خاصية العمل عبر أنظمة التشغيل المختلفة وذلك لواجهات الاستخدام أو الوسائط Interfaces الخاصة بالبرمجة من خلال المواصفة المعيارية W3C DOM Level 2. هذا ولقد تم تصميم هذه الوحدة الجديدة بحيث يكون لها بناء هيكل أو بناء معمارى مفتوح مما يؤدى إلى جعل من السهولة بمكان تكامل الـInterfaces مع النموذج DOM المعتمد على الحلول المعدة بلغة XML.

استخدام XML فى تطبيقات قواعد البيانات

هناك مجموعة جديدة من المكونات تمكنك من تكامل المستندات المعدة بلغة XML داخل الهيكل المعمارى لتطبيقات قواعد البيانات. ومثل هذه المكونات الجديدة تجعل من الممكن استخدام ملفات التحويل المكونة بواسطة XML Mapper وهى عبارة عن أداة يتم استخدامها فى مرحلة التصميم لتعريف وتحديد خرائط Mappings تربط بين أى مستند مصمم بلغة XML وبين حزمة البيانات لأى تطبيق من تطبيقات قواعد البيانات.

المظاهر الجديد لمترجم اللغة Compiler

المتغيرات Variants

كافة الروتينات المختصة بمعالجة المتغيرات تم نقلها من وحدة النظام System Unit إلى وحدة جديدة تسمى Variants. وبالرغم من إنه فى إمكانك حتى الآن استخدام أنواع المتغيرات بدون اشتمال أو ضم الوحدة الجديدة Variants داخل التطبيقات التى تتولى إعدادها إلا إن المتغيرات الخاصة بالتعامل مع مجموعة الدوال RTL تكون موجودة بالوحدة الجديدة Variants. هذا ولكى تستخدم هذه الدوال فسوف تحتاج لضم الوحدة الجديدة Variants فى جملة الاستخدامات بالتطبيق الذى تتولى إعداده.



تستطيع الآن تعريف أنواع بيانات خاصة أو مفصلة للمتغيرات. كما أن المتغيرات أصبحت الآن تعمل على تدعيم Int64.

أنواع البيانات الرقمية أو العددية Enumerated

أصبح الآن فى الإمكان تخصيص قيمة معينة للعددات Enumerations كما هو موضح من خلال المثال التالى :

```
type
        TInfo = (iZero, iOne, iTwo, iFour = 4);
```

التغييرات التى أجريت على وحدة الثوابت Consts Unit

وحدة الثوابت Consts.pas تم تقسيمها إلى ملفين هما : الملف Consts.pas والملف RTLConsts.pas.

الموجهات directives الجديدة لدى مترجم اللغة

هناك عدد من الموجهات directives الجديدة تم توثيقها فى الدليل الخاص بلغة pascal. ونود هنا القول بأن شركة مايكروسوفت يزداد إعتمادها يوما بعد يوم على أعلام المقدمة header flags الخاصة بـ PE (اختصار للمصطلح Portable executable والذى يعنى قابلية التنفيذ المتنقلة) وذلك للسماح لأى تطبيق بأن يشير بالتوافق مع خدمات نظام التشغيل OS (Operating System) أو يطلب المزيد من خدمات نظام التشغيل المتقدمة. وفى هذا الصدد نقول أن مترجم اللغة بـ Delphi 6 أصبح الآن يعمل على تدعيم إثنين من الموجهات الجديدة والتى تعمل على توفير العديد من الخيارات القوية لجعل التطبيقات المعدة بلغة Delphi تعمل بكفاءة عالية فى أنظمة تشغيل ويندوز NT و٢٠٠٠. وهذين الموجهين هما :

```
{$SetPEFlags <integer expression>}
```

و

```
{$Set PEFlags <integer expression>}
```



هذه الموجهات معدة للمبرمجين المحترفين فقط.

الموجهات الشرطية Conditional directives

لـو أنـك تكتـب تطبـيقات لكـى تعمل فى بيئات الويندوز المختلفة فى هذه الحالة تسـتطيع أن تسـتخدم الرمز MSWINDOWS لتجعل التطبيقات التى تتولى إعدادها لديها القـدرة عـلى الشـعور بوجود بيئة الويندوز وإلا ستشعر بأنها تعمل فى بيئات تشغيل أخرى مثل Linux.

المثال التالى يوضح لنا كيفية استخدام الرمز السالف الذكر :

```
{$IFDEF MSWINDOWS}
        Get Desperate;
{$ENDIF}
```

الموجهة $If

هناك موجهة جديد وهو الموجهة $If الذى يقوم بتقييم تعبيير ثابت كما هو موضح من خلال المثال التالى :

```
{$IF Defined(WIN32) and (SomeConst > 12.0) }
...
{$ENDIF}
```

والجديد فى الموضوع أنه يمكن الآن تقييم أدوات تعريف ثوابت الباسكال من خلال الموجهـات $IF. ونـود هـنا القول بأنه يمكن اختيار وجود رمز للتعريف الشرطى $IFDEF وذلـك مـن خـلال الدالـة Defined() كمـا إنـه يمكن اختبار وجود رمز لأداة تعريف لثابت الباسكال من خلال الدالة Declared().

المثال التالى يوضح لنا ما سبق :

```
{$IF  Defined(WIN32) and Declared(MyConst)}
...
{$ENDIF}
```

حقل الضبط $ALIGN

الآن أصبح لدى حقل الضبط $ALIGN الخيارات الجديدة التالية :

- {$A1}
- {$A2}
- {$A4}
- {$A8}

أداة التجميع الأساسية built-in assembler الجديدة

الآن أصبح Delphi 6 مشـتملا عـلى أداة تجميع أساسـية built-in assembler جديدة تماما تتميز بالآتى :

- الموجهات الجديدة VMTOFFSET وDMTINDEX.
- تدعيم التوجية instruction الجديد لكل من MMX وSIMD و Enhanced MMX وIntel SSE لكل من وحدات المعالجة المركزية CPU التى من الطراز Pentium Pro وPIII وP4 وكذلك التى من الطراز AMD Enhanced 3D الخاص بـAMD K7.
- تقديم الدعم الجديد للبيانات DQ (اختصار للمصطلح Define Quadword).

التحديد الافتراضى الجديد للثوابت القابلة للتحديد

التحديد الإفتراضى لموجهة المترجم $WRITEABLECONST قد تم تغييره من ON إلى OFF. وهذا يعنى أنه يجب عليك بشكل صريح تشغيل هذا الموجهة قبل أن تتمكن من التعامل مع القيم التى تنتمى لنوع البيانات Const.

تقديم الدعم لأنواع الملفات المختلفة

من أجل زيادة الدعم الخاص بإمكانية إعداد تطبيقات تكون لديها القدرة على العمل عبر مختلف أنظمة التشغيل نجد أن مترجم اللغة أصبح قادرا على التعامل مع الملفات النصية التى تتبع نمط نظام التشغيل Linux. ومعنى هذا أن الملفات النصية اصبحت تشتمل على سطور تنتهى بالحرف المناظر للأسكى كود 10 بدلا من أن تنتهى بالحرف المناظر للأسكى كود 13 كما هو الحال بالملفات النصية المستخدمة فى بيئة الويندوز.

التغييرات التى أجريت على وظيفة التحميل الزائد Overload

العملية التى يقررها المترجم الخاصة بلغة Delphi (والتى تؤدى إلى حدوث تحميل زائد على مترجم اللغة من أجل العوثر على أفضل معامل من ضمن مجموعة من المعاملات) قد تم تحديثها فى Delphi 6 لقديم الدعم للحالات التالية والتى كان يعتبرها المترجم قبل ذلك حالات ميئوس منها تماما :

- أصبح الآن لدى مترجم Delphi 6 القدرة على تمييز دوال (أو وظائف) التحميل الزائد والتى تشتمل على AnsiString/PChar عن تلك التى تشتمل على المعاملات WideString/WideChar فى نفس موضع المعامل.
- من الممكن الآن جعل المتغيرات Variants كما لو كانت معاملات فى جمل الإعلانات عن دوال التحميل الزائد. ونود هنا القول بأن أى Variant يمكن

اعتباره أكثر عمومية من أى نوع بيانات بسيط.

- عملية تمرير حالات الكائن إلى دوال التحميل الزائد من خلال نسخة من الكائن ونسخة من الـInterface لم تعد تسبب أى مشاكل لمترجم اللغة.
- أصبح من الممكن الآن تقبل NIL على أساس أنها قيمة صحيحة لأى من المعاملات الخاصة بنوع الـInterface داخل أى من دوال التحميل الزائد أو الفوقى.

المظاهر الجديد لأدوات التحكم ActiveX والنموذج COM (الإصدار الفنى Professional والإصدار المتكامل Enterprise)

هناك العديد من المظاهر والإمكانيات التى تم إضافتها لكل من أدوات التحكم ActiveX والنموذج COM نذكر منها ما يلى.

التسجيل والتركيب لصفات تهيئة النموذج COM

تستطيع الآن تخصيص صفات النموذج +COM للكائنات COM الجديدة.

المعالج Event Object Wizard

الشكل التالى يوضح لنا المعالج الجديد COM+event Wizard :

شكل توضيحى :

COM+ Event Object

CoClass Name

Interface

Description

OK Cancel Help

وهذا المعالج يسمح لك بأن تنشأ كائنات COM+event.



ينبغى عليك حتى الآن إضافة كود برمجى بنفسك لجعل الأحداث تقع أو لجعل كود برمجى معين يستجيب للأحداث التى تقع.



تنفيذ الـInterfaces الموجودة بالفعل

الشكل التالى يوضح لنا المعالج COM Object Wizard:

شكل توضيحى :

COM Object Wizard

Class Name:
Instancing: Multiple Instance
Threading Model: Apartment
Implemented Interface: List
Description:
Options
☑ Include Type Library ☑ Mark interface Oleautomation
OK Cancel Help

الآن تستطيع أن تستخدم هذا المعالج لتكوين كائن خادم من أجل إعداد Interface افتراضية تكون منتمية لمكتبة الأنواع المسجلة لدى نظام التشغيل الذى تتعامل معه. وفى السابق كان هذا المعالج يقوم دائما بتنفيذ أى Interface يتم إنشاؤه حديثا (نزولا من IUnknown). والآن نجد أن المعالج COM Object Wizard يسمح لك أيضا بأن تختار Interface من أى مكتبة من مكتبات الأنواع المسجلة لدى النظام.

بالإضافة لما سبق نجد أن المعالج COM Object Wizard يقوم بإنشاء معلومة مكتبة النواع من أجل القطاع CoClass لكى يتم تنفيذ هذا الـInterface وفى نفس الوقت يتم استخدام قطاع من قطاعات التنفيذ مع عدد من الطرق الهيكيلة (بدلا من أن تقوم بذلك بنفسك) لاستكمال عملية التنفيذ.



قطاع التنفيذ يتوارث كل من طريقة التنفيذ **IUnknown** وطريقة التنفيذ **IDispatch**.

الكائنات الحركية Transactional (كبديل للمعالج MTS)

أصبح فى الإمكان الآن إنشاء الكائنات الحركية Transactional Objects باستخدام الاصدار الفنى Professional للغة Delphi 6. وفى السابق كان الدعم MTS قاصرا على الإصدار المتكامل Enterprise فقط.

الدعم +MTS/COM المزدوج للكائنات الحركية

معالج الكائن MTS تم استبادلة بمعالج الكائن الحركي الموضح في الشكل التالي :

شكل توضيحي :

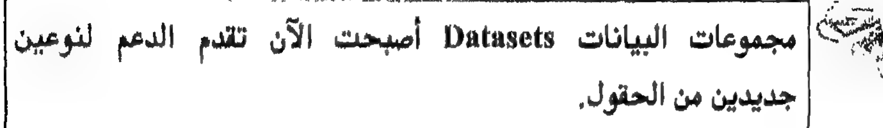
New Transactional Object
CoClass Name
Threading Model: Apartment
Transaction model
Requires a transaction
Requires a new transaction
Supports transactions
Does not support transactions
Ignores Transactions
Options
Generate Event support code
OK Cancel Help

هـذا المعـالج الجديد لدية القدرة على إنشاء كائنات يمكن استخدامها مع COM+ أو مع MTS.

المظاهر الجديدة لقواعد البيانات
(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

يشـتمل الإصـدار السادس من لغة Delphi على آلية جديدة للوصول للبيانات وهى تعـرف بــdbExpress وهـذه الآلـية تعمـل عـلى توفـير وسـيلة سريعة للغاية وسهلة للغاية للوصول لخوادم قواعد البيانات SQL وذلك بأستخدام إطار هيكلى Framework يجعل من السهولة بمكان كتابة مشغلات قواعد بيانات تنتمى للطائفة Third-party.



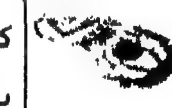

مجموعات البيانات Datasets أصبحت الآن تقدم الدعم لنوعين جديدين من الحقول.

لقـد تم إضافة عدد المكونات الجديدة إلى Delphi 6 لجعل من الأسهل التعامل مع مجموعـات بيانات العملاء Client Datasets وذلك من أجل إعداد تطبيقات قواعد بيانات ذات طبقتين two-tier أو ذات طبقات متعددة Multi-tier.

آلية الوصول للبيانات dbExpress

آلية الوصول للبيانات dbExpress عبارة عن مجموعة من مشغلات قواعد البيانات البسـيطة lightweight تعمـل عـلى توفـير وسـيلة سـريعة للوصـول لخودام قواعد البيانات

SQL. وبالنسبه لكل قاعدة بيانات تم تدعيمها نجد أن الآلية dbExpress تعمل على توفير مشغل Derive تعمل على ضبط برنامج لخادم معين لإعداد interfaces متجانسة للآلية dbExpress. وعندما تقوم بنشر وتوزيع التطبيق الذى تتولى إعداده تحتاج لأن تضم إما ملف DLL واحد فقط (خاصة بمشغل خادم معين) مع ملفات التطبيق الذى تقوم ببناؤه مع اختيار عدد من الملفات الإضافية التى تسمح لك بأن تقوم بتحميل معلومات عن الاتصال مع الخادم المعنى فى أثناء مرحلة التشغيل Run-Time.



كذلك تستطيع نشر وتوزيع التطبيق الذى تتولى إعداده على أساس كونه برنامج تنفيذى EXE مستقل بذاته Standalone.

بناء على الاصدار الذى تستخدمه من لغة Delphi 6 سيكون Delphi مشتملا على المشغلات التالية والتى تخص الآلية dbExpress :

المشغل	اسم الملف
InterBase	DBEXPINT.DLL
DB2	DBEXPDB2.DLL
Oracle	DBEXPORA.DLL
MySQL	DBEXPMYS.DLL

المكونات الأساسية والمعدة خصيصا من أجل التعامل مع المشغلات الخاصة بالآلية dbExpress عبارة عن الآتى :

- المكون TSQLConnection.
- المكون TSQLDataSet وهو عبارة عن مجموعة بيانات إحادية الإتجاه unidirectional.
- كل من المكون TSQLQuery والمكون TSQLStoredProc والمكون TSQLTable وهى عبارة عن مجموعات بيانات تتصف بإنها إحادية الإتجاه unidirectional وفى نفس الوقت متوافقة مع العناصر الموجودة بالفعل وذلك فى حالة أنك ترغب فى port اى تطبيق.

مجموعـات البـيانات الـتى تسـتخدم الآلـية dbExpress يطلـق علـيها مجموعات بـيانات إحاديـة الإتجـاه unidirectional وذلـك لأن الآلية dbExpress تقوم بتنفيذ مؤشر إحـادى الإتجـاه فقـط للوصـول للسـجلات. وفـى هـذا الصـدد نقول إن مجموعات البيانات الإحادية الإتجاه لا تحتفظ بالبيانات بالذاكرة العشوائية RAM من خلال الخاصية buffer ممـا يجعلهم أسرع وأقل استهلاكا لمصادر النظام من الأنواع الأخرى من مجموعات البيانات ولكن فى نفس الوقت تؤدى إلى وجود العديد من القيود.

فعـلى سـبيل المـثال لا تسـتطيع أن تجعـل مجموعـة بـيانات إحادية الإتجاه تقيم اتصـال مـع شـبكة بـيانات (مجموعـة بـيانات ثنائـية الاتجـاه) وذلـك لأنه ليس هناك أى سجلات مخزنة بذاكرة النظام.

العديـد مـن القدرات والإمكانيات الخاصة بمجموعة البيانات TDataSet تكون إما غير قابلة للتنفيذ داخل مجموعات البيانات الإحادية الاتجاه أو تجعل تصرفاتهم خارج أى توقعـات ممكـنة. هذا وبالرغم من القيود المفروضة على مجموعات البيانات الإحادية الإتجاه إلا إنـه يمكـن اعتـبار هـذه النوعـية مـن مجموعات البيانات وسيلة قوية للوصول للبيانات. فهى تتميز بالسرعة والبساطة الشديدة سواء عند الاستخدام أو عند النشر والتوزيع.

لكـى تقـوم بـتحرير بـيانات مـن مجموعـات بيانات إحادية الإتجاه عليك أن تقيم اتصال بينهم وبين مجموعة بيانات عميلة client dataset. ونود هنا القول بأن الاتصال بين مجموعـة البـيانات العمـيلة ومجموعة البيانات الإحادية الإتجاه يتم من خلال مصدر توفير مجموعات البيانات.

أنواع الحقول الجديدة

لقـد تم إضـافة نوع الحقل الجديد TFMTBCDField إلى Delphi 6. وهذا النوع الجديـد يمـثل تكويـد ثنائى عشرى BCD (اختصار للمصطلح Binary-Coded Decimal) حقـيقى فى مقابل نوع الحقل TBCDField الموجود بالفعل والذى يعمل على تحويل القيم BCD لتصبح تابعـة لـلنوع Currency. وأنت تستطيع توصيف نوع الحقل TBCDField عـلى أسـاس كونـه حقـل رئيسـى لحقـل BCD مـن أى مجموعـة بيانات. هذا ومن خلال مجموعـات البـيانات الخاصـة بالآلـية dbExpress نجـد أنـه يـتم إنشاء مكونات الحقول الديناميكـية الـتى تنـتمى لـنوع الحقـل TBCDField وذلـك عـند اسـتخدام نـوع الحقـل TBCDField قد ينتج عنه فقد فى دقة القيم precision.





فى الواقع يمكن القول بأن ما سبق يمكن أن يمتد لكى يشمل كل من مجموعات البيانات BDE, ADO وإحتمال أن يطول مجموعات البيانات IBX.



بالدعم الخاص بنوع الحقل TBCDField نجد أن الروتينات العامة CurrToBCD وBCDToCurr قد تم نقلهما من الوحدة db إلى الوحدة FMTBCD.

هناك نوع جديد من أنواع الحقول الجديدة المضافه إلى Delphi 6 وهو نوع الحقل TSQLTimeStampField الذى يعمل على تدعيم تمثيل القيم التاريخية والزمنية Date/Time المستخدمة فى المشغلات الخاصة بالآلية dbExpress.

الهياكل البنائية لمجموعة بيانات العميل Client Dataset

يشتمل الإصدار السادس من لغة Delphi ثلاثة مجموعات بيانات عملاء جديدة وهى تضم مجموعة بيانات مصدرية وموفر أساسى Built-in Provider. ومجموعات البيانات هذه الغرض الأساسى منها تبسيط عملية استخدام موفر ومجموعة بيانات عميلة للإلتقاط التحديثات بالتطبيقات البسيطة جدا.

فيما يلى مجموعات البيانات العميلة الجديدة :

- مجموعة البيانات TBDEClientDataSet.
- مجموعة البيانات TSQLClientDataSet.
- مجموعة البيانات TIBClientDataSet.

هناك قطاع تأسيس مشترك جديد يسمى TCustomClientDataSet أصبح الآن موجودا لمجموعة البيانات TClientDataset ومجموعات البيانات الثلاثة الجديدة السالفة الذكر.



هذه المكونات مصممة من أجل التطبيقات البسيطة فقط. وعندما يستخدم التطبيق العلاقات التبادلية التفصيلية أو يحتاج للحصول على بيانات من خادم عدة مرات فى هذه الحالة نجد أن مستوى أداء يصبح أفضل خاصة عند استخدام موفر خارجى ومجموعة بيانات عميلة.

مجموعة البيانات TClientDataSet لديها العديد من الخصائص الجديدة التالية :

الخاصية ConnectionBroker :

هذه الخاصية تسمح لك بأن تضيف طبقة زيادة تعمل على إعادة التوجية إلى مواصفات أى من مكونات الإتصال. ومثل هذه الخاصية تكون مهمة بصفة خاصة عندما ترغب فى استخدام نوع من أنواع الإتصال فى مرحلة التصميم ونوع آخر من أنواع الإتصال عندما تقوم بنشر وتوزيع التطبيق الذى تتولى إعداده. ولو أن لديك العديد من مجموعات البيانات العميلة داخل أحد التطبيقات وكانت كلها تستخدم نفس الخاصية ConnectionBroker فى هذه الحالة تستطيع تغيير المكون الذى يصل بينهم كلهم وبين خادم التطبيق وذلك عن طريق تغيير خاصية واحدة فقط (الخاصية Connection الخاصة بأداة كسر أو قطع الإتصال) وذلك بدلا من الحاجة إلى تغيير الخاصية RemoteServer لكل مجموعة بيانات عميلة بالتطبيق.

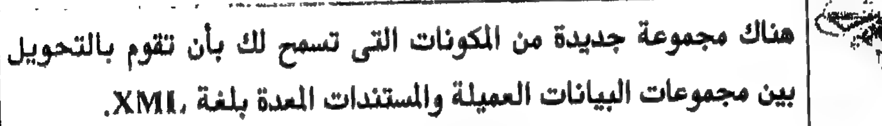
الخاصية DisableStringTrim :

هذه الخاصية تسمح لك بأن تتحكم فى طريقة تعامل مجموعة بيانات العميلة للمسافات الموجودة بالقيم التى يدخلها المستخدم بالحقول.

الخاصية XMLData :

هذه الخاصية تمنحك القدرة على الوصول لحزمة البيانات الخاصة بمجموعة البيانات العميلة علما بأن هذه البيانات معدة بلغة XML.

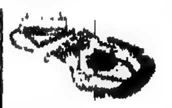




هناك مجموعة جديدة من المكونات التى تسمح لك بأن تقوم بالتحويل بين مجموعات البيانات العميلة والمستندات المعدة بلغة XML.

لقد تم تحسين وتطوير المكون TUpdateSQL بحيث تستطيع الآن أن تستخدم العديد من كائنات التحديث عند التقاط التحديثات باستخدام مجموعة بيانات عميلة. ونود هنا القول بأن الخاصية DataSet الخاصة بالمكون TUpdateSQL قد تغيرت من كونها تتطلب TBDEDataSet إلى كونها تتطلب TDataSet فقط. هذا وعند استخدام مجموعة بيانات عميلة والعديد من كائنات التحديث فإنه فى هذه الحالة ينبغى عليك تحديد قيمة

هــذه الخاصــية مــن خــلال المعــامل DeltaDS الخــاص بــأداة معاملــة الحــدث BeforeUpdateRecord الخاصــة بالموفــر. وفــى هــذه الحالــة لــن يــتمكن المكــون TUpdateSQL من معرفة اسم قاعدة البيانات والـsession من الخاصية DataSet الخاصة بهـذا المكـون كمـا إنـه ينـبغى علـيك أيضـا تحديـد قـيمة كـل مـن الخاصـية الجديـدة DatabaseName والخاصية الجديدة SessionName.



المكون TUpdateSQL يعمل بنفس الطريقة التى كان يعمل بها فى الإصدارات السابقة للغة Delphi وذلك عندما يكون هناك كائن تحديث واحد فقط أو عند استخدام مجموعة البيانات BDE (المتاحة للعمل) للإلتقاط التحديثات التى تتم.

الدعم الخاص بالتطبيقات المتعددة الطبقات multi-tiered

أصبح مـن المـتاح الآن مكونين إتصال جديدين وهما يسمحان لك بأن تعمل بمرونة أكـبر مـع مجموعـات البـيانات العمـيلة بالتطبيقات المتعددة الطبقات multi-tiered. وهذه المكونين الجديدين هما :

مكون الإتصال المشترك TSharedConnection :

مكـون الإتصـال هـذا يسـمح للتطبـيق العمـيل بـأن يعمـل عـلى تكويـن العديـد من الإتصـالات بـالعديد مـن Modules البـيانات البعيدة الموجودة بتطبيق خادم واحد فقط. هذا ومن خلال استخدام المكونات TSharedConnection يؤدى إلى أن كافة الاتصـالات بـــModules البـيانات البعـيدة تسـتخدم اتصال واحد مشترك بتطبيق الخـادم ممـا يسـمح لتطبـيق الخـادم بـأن يشـعر بهـذه الكائـنات وهذه الإتصالات ومصادرها أيضا.

مكون الإتصال المحلى TLocalConnection:

هذا المكون يعمل كما لو كان مكون إتصال بالنسبة لموفرى البيانات المحليين (فى نفس التطبـيق عـلى أساس أنه مجموعة بيانات عميلة). ومن خلال استخدام مكون الإتصال المحلى TLocalConnection تستطيع أن تجعل استخدام الوسيط IAPPServer يتم

بشكل صريح بالإضافة إلى إمكانية تبسيط عملية الـScaling Up التى تتم لاحقا من أجل استخدام موفر بيانات بعيد بأى تطبيق من تطبيقات الخادم.

بالإضافة إلى ما سبق يوجد كائن جديد يسمى TPacketInterceptFactory وهو يجعل من الأسهل اعتراض الرسائل المتبادلة بين تطبيق العميل وتطبيق الخادم وذلك عند استخدام Sockets للإتصال. هذا ومن خلال استخدام الكائن الجديد TPacketInterceptFactory يتم تلقائيا تسجيل كائن الاعتراض ومن ثم تستطيع تخصيص مثل هذا الكائن إلى المكون الذى يعمل كـSocket الاتصال وذلك باستخدام قائمة منسدلة موجودة فى نافذة أداة فحص الكائن Object Inspector.

التغييرات التى أجريت على باللية المكونات

باللية المكونات قد تم تغييرها بحيث تعكس النمو الهائل فى الخيارات المختلفة والمتعددة التى أصبحت متاحة عند التعامل مع تطبيقات قواعد البيانات وتطبيقات الإنترنت.

- باللية المكونات قد تم إعداة تنظيمها للتأكيد على أن مولد قاعدة البيانات Borland Database Engine الذى أعددته شركة بورلاند (BDE) يعتبر من أقوى مولدات قواعد البيانات الموجودة الآن علما بأن هذا المولد يعتبر بديل للألية الأساسية الوصول للبيانات. وفى هذا الصدد نقول أن المكونات المعتمدة على مولد قاعدة البيانات BDE قد تم نقلها إلى التبويب BDE Page بباللية المكونات كما هو موضح بالشكل التالى :

شكل توضيحى :

System | Data Access | Data Controls | dbExpress | DataSnap BDE | ADO

- لقد تم تحديث مجموعة المكونات InterBaseExpress وهذا التحديث اشتمل تحسين وتطوير الدعم المقدم للأحداث المرتبطة بهذه المكونات وكذلك الدعم الخاص بأدوات التكوين InterBase والتى تقوم تلقائيا بتكوين قيم الحقول. ومن ثم فقد تم إضافة صفحة جديدة بباللية المكونات وهى الصفحة InterBase Admin الموضح فى الشكل التالى :

شكل توضيحى :

Data Controls | dbExpress | DataSnap | BDE | ADO | InterBase | WebServic

- لقد تم إضافة الصفحة الجديدة dbExpress إلى باللية المكونات. وهذه الصفحة الجديدة تشتمل على المكونات dbExpress التى سبق التعرض لها فى هذا الفصل :

شكل توضيحى :

Data Controls | dbExpress | DataSnap | BDE | ADO | InterBase | WebServic

- لقد تم نقل الصفحة Midas كما تم تغيير استخدام المصطلح MIDAS. فكل من مجموعة بيانات العميل ومجموعة بيانات الموفر التى يتم استخدامهما ليكونا فى الصفحة Midas قد تم نقلهما إلى الصفحة Data Access الموضح فى الشكل التالى :

شكل توضيحى :

Win32 | System | Data Access | Data Controls | dbExpress | DataSnap | BDE

- كل مكونات الإتصال التى يتم استخدامها للاتصال بأى تطبيق من تطبيقات الخادم وأداة تقسيم الكائن البسيط قد تم نقلهما إلى الصفحة الجديدة DataSnap كما هو موضح بالشكل التالى :

شكل توضيحى :

Win32 | System | Data Access | Data Controls | dbExpress | DataSnap | BDE

- الصفحة الجديدة DataSnap أصبحت الآن مشتملة على المكونات لمستخدمة لإتصال مجموعات بيانات العملاء بأى من تطبيقات الخادم. وهذه المكونات تضم مكونات الاتصال وأداة تقسيم الكائن البسيط والتى كانت موجودة قبل ذلك بالصفحة Midas. بالإضافة لذلك تضم الصفحة DataSnap ثلاثة مكونات جديدة وهى :
- المكون TConnectionBroker وهو يعمل كما لو كان مكون وسيط بين مجموعة بيانات العميل وأى مكون من مكونات الإتصال.

- المكون TSharedConnection وهو يتصل بأى تطبيق من تطبيقات الخادم على أساس كونه شريك فى العديد من Modules البيانات البعيدة.
- المكون TLocalConnection وهو يمثل الاتصال بموفرى البيانات التى تكون فى نفس التطبيق مثل أى مجموعة من مجموعات بيانات العميل.

المظاهر والإمكانيات الجديد لـCORBA (الإصدار المتكامل Enterprise فقط)

بعض من الإصدارات الخاصة بلغة Delphi تأتى وهى مزودة بالمترجم IDL2PAS وذلك لكتابة نوعية خاصة من التطبيقات تعرف بتطبيقات الـCORBA. ولقد تم تطوير وتحسين المترجم IDL2PAS بحيث أصبحت لدية القدرة على تدعيم عملية كتابة تطبيقات لخوادم الـCORBA (تكوين كود برمجى هيكلى) بالإضافة إلى كتابة كود برمجى Stub يعمل بتطبيقات العملاء المتصلين بالخوادم الـCORBA.

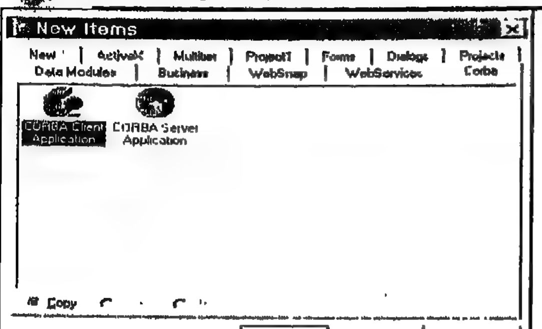
هذا وتستطيع الوصول للمترجم IDL2PAS عن طريق فتح القائمة Tools ثم اختيار الأمر Regenerate CORBA IDL Files ليظهر على الشاشة صندوق الحوار Regenerate CORBA IDL Files الموضح فى الشكل التالى :

شكل توضيحى :

Regenerate IDL Files
Application | Options
Application Type
Console Application
Windows Application
Add IDL Files
Add
Cancel

أو عن طريق فتح القائمة File ثم اختيار New ثم Other وبصندوق الحوار New Items انقر بالماوس على التبويب CORBA حتى يظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :



المترجم السالف الذكر يكون مفيدا بالنسبة لعمليات إنشاء كل من تطبيقات العملاء وتطبيقات الخادم بالإضافة إلى صيانة المشاريع CORBA التى سبق إعدادها حيث يمكن أن تجرى تغيير على الملف IDL ثم إنعاش Refresh ملفات المشروع لجعلهم يشعروا بالتغيير الذى أجريته.

من الأشياء التى ننصح بها بشدة هو أن تستخدم المترجم الجديد IDL2PAS الخاص بتطبيقات الـCORBA وذلك بدلا من استخدام دعم CORBA القديم الذى كان متكاملا مع الدعم الذى تقدمه لغة Delphi للتطبيقات COM. هذا وعند استخدام المترجم الجديد IDL2PAS فى هذه الحالة ينبغى عليك استخدام الوحدة الموجودة بالملف corba.pas والوحدة الموجودة بالملف orbpas30.pas أو تستخدم الوحدة الموجودة بالملف orbpas40.pas للتفاعل مع ORB بدلا من الوحدات القديمة الموجودة بكل من الملف corbaobj.pas والملف orbpas.pas. بالإضافة للوحدات السالفة الذكر فإنه توجد إثنين من الوحدات الجديدة وهما :

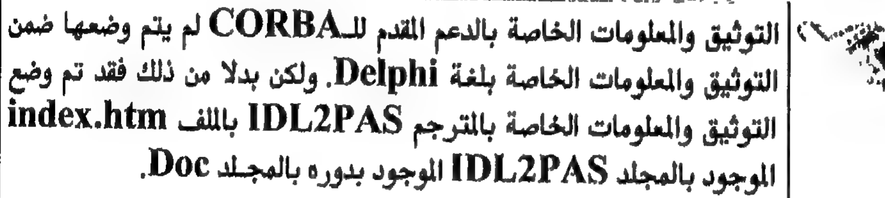
الوحدة cosevent.pas :

هذه الوحدة تعمل على تقديم خدمات للأحداث التى تقع للمكونات والوحدة cosnaming.pas.

الوحدة cosnaming.pas :

هذه الوحدة تعمل على تقديم خدمات للتسمية.




التوثيق والمعلومات الخاصة بالدعم المقدم للـCORBA لم يتم وضعها ضمن التوثيق والمعلومات الخاصة بلغة Delphi. ولكن بدلا من ذلك فقد تم وضع التوثيق والمعلومات الخاصة بالمترجم IDL2PAS بالملف index.htm الموجود بالمجلد IDL2PAS الموجود بدوره بالمجلد Doc.

مظاهر وإمكانيات الأفعال Actions الجديدة
(الإصدار الفنى Professional والإصدار المتكامل Enterprise)

فيما يلى سنتعرف سويا على المظاهر والإمكانيات التى تتمتع بها الأفعال Actions المضافة حديثا للإصدار السادس من لغة Delphi.

حزم الأفعال Action Bands

من السهولة بمكان التعامل مع الأفعال فهى تعمل على تبسيط عملية إعداد وتطوير واجهة الاستخدام للتطبيقات التى تتولى إعدادها وذلك عن طريق استخدام مجموعة من الأدوات الجديدة التى تعرف فى مجموعها بحزم الأفعال ActionBands. وأنت تستطيع تنظيم الأفعال والصور كما تستطيع إضافتهم إلى القوائم وشرائط الأدوات التى تتولى تفسيلها بنفسك. وهذه الأدوات يمكن الوصول إليها من خلال الصفحة Additional بباللينة المكونات كما هو موضح بالشكل التالى :

شكل توضيحى :

فيما يلى سوف نستعرض سويا مجموعة الأدوات الموجودة بالصفحة Additional :

الأداة TActionManager :

الشكل التالى يوضح لنا مدير الأفعال Action Manager :

شكل توضيحى :

وهذا المدير يتولى مهمة إدارة قوائم الأفعال وذلك من أجل تنظيم الأفعال سواء كانت مفصلة أو قياسية.

الأداة TActionMainMenuBar و الأداة TActionToolBar :

مـن خـلال الأداة TActionMainMenuBar تسـتطيع أن تقوم بتفصيل قوائم Menus كما أن الأداة TActionToolBar تعمل على تفصيل شرائط أدوات Toolbars وبكل من هذه القوائم وشرائط الأدوات تستطيع أن تقوم بسحب وإسقاط الأفعال أو الأوامر وذلك من محـرر مديـر الأفعال. كما تستطيع أيضا سحب وإسقاط مجموعة من الأفعال التى تنتمى لـنوع معـين (تحصل على كافة الأفعال الموجودة ضمن أى قائمة فرعية) أو تقوم بسحب وإسقاط فعل معين داخل أى قائمة أو شريط أدوات.

الأداة TCustomizeDlg :

مـن خلال هذه الأداة يتم فتح صندوق حوار التفصيل Customize وهو يعمل بنفس الطـريقة الـتى يعمـل بهـا محـرر مديـر الأفعال ولكنه مخصص لمستخدمى التطبيق الـذى تتولى إعداده وذلك لكى يتمكنوا من تعديل محتويات القوائم وشرائط الأدوات فـى أثناء مرحلة تشغيل التطبيق. وفى هذا الصدد نقول إن المكون المتمثل فى الأداة TCustomizeDlg يعمـل بالتوافق مع الأداة TActionManager. وأنت تستطيع إمـا أن تحـدد قـيمة الخاصـية ActionManager لـلمكون ActionManager أو تضـيف فعل قياسى (TCustomizeActionBars) إلى محرر مدير الأفعال وإلى أى من القوائم التى يتم إعدادها من خلال الأداة TActionMainMenuBar.

تشـتمل لغـة Delphi عـلى مجموعـة مـن الأفعـال القياسية الجديدة والتى يمكنك إضـافتها للقوائـم وشـرائط الأدوات الـتى تـتولى إعدادهـا مـن خـلال كـل مـن الأداة TActionMainMenuBar والأداة TActionToolBar. ونـود هـنا القـول بـأن قطاعـات الأفعال المعرفة مسبقا تضم كل من الآتى :

- Format
- File
- Search
- Tab
- List
- Dialog
- Internet
- Tools

تستطيع أيضا إعداد وتهيئة القوائم التى تقوم بتفصيلها لكى تتمكن من إخفاء أو إظهار الأفعال بناء على تتابع استخدام كل منها. وللقيام بذلك عليك أن تستخدم الخاصية HideUnused للعديد من الكائنات TActionClient والتى تعمل على وضع العناصر الأقل استخداما فى القائمة داخل قائمة فرعية يمكن الوصول إليها عن طريق وضع مؤشر الماوس على سهم مزدوج بنهاية القائمة أو شريط الأدوات.

الأفعال القياسية الجديدة

الأفعال القياسية التالية تم إضافتها حديثا إلى VCL :

أفعال التنسيق Format:

- TRichEditBold
- TRichEditItalic
- TRichEditUnderline
- TRichEditStrikeOut
- TRichEditBullets
- TRichEditAlignLeft
- TRichEditAlignRight
- TRichEditAlignCenter

الأفعال الخاصة بنظام المساعدة Help :

- THelpContextAction

الأفعال الخاصة بالتعامل مع الملفات :

- TFileOpen
- TFileSaveAs
- TFilePrintSetup
- TFileRun
- TFileExit

الأفعال الخاصة بعملية البحث :

- TSearchFind
- TSearchFindFirst
- TSearchReplace
- TSearchFindNext

الأفعال الخاصة بالتبويب Tab (أداة التحكم Page) :

- TPreviousTab
- TNextTab

الأفعال الخاصة بقوائم العرض List Actions :

- TListControlCopySelection
- TListControlDeleteSelection
- TListControlSelectAll
- TListControlClearSelection
- TListControlMoveSelection
- TStaticListAction
- TVirtualListAction

الأفعال الخاصة بصناديق الحوار Dialog Actions :

- TOpenPicture
- TSavePicture
- TColorSelect
- TFontEdit
- TPrintDlg

الأفعال الخاصة بالإنترنت Internet Actions :

- TBrowseURL
- TDownLoadURL
- TSendMail

الأفعال الخاصة بالأدوات Tool Actions :

- TCustomizeActionBars

التحسينات التى أضيفت لقطاعات الأفعال

لقد تم تطوير وتحسين TCustomAction بحيث أصبحت تمتلك الخصائص الجديد التالية :

- GroupIndex
- SecondaryShortCuts
- HelpKeyword
- HelpType
- AutoCheck

أما TCustomActionList فقد أصبحت تمتلك الآن الخاصية State التى تسمح لك بأن تقوم بشكل مؤقت بتعطيل كافة الأفعال الموجودة بقائمة العرض List.

المظاهر والإمكانيات الجديدة للوحدات VCL الجديدة (كافة الإصدارات)

فيما يلى سوف نستعرض سويا المظاهر والإمكانيات الجديدة للوحدات VCL الجديدة بالإصدار السادس من لغة Delphi.

المكونات الجديدة

فيما يلى سنتعرف سويا على المكونات الجديدة الخاصة بالوحدات VCL الجديدة.

المكون TLabeledEdit :

هذا المكون يعتبر فى حد ذاته إضافة للوحدة ExtCtrl.pas وهو يعمل على توضيح كيفية استخدام المكونات الفرعية. وهو عبارة عن أداة تحكم للتحرير Edit تمتلك Label ملحق بها. ونود هنا القول بأن الـLabel يظهر على أساس كونه خاصية لأداة التحكم هذه.

المكون TValueListEditor :

هذا المكون عبارة عن شبكة مفصلة أو خاصة يتم استخدامها لتحرير المكونات TStrings التى تشتمل على أزواج من مفاتيح لوحة المفاتيح والقيم المناظرة لها. وهذا المكون يعمل بنفس الطريقة التى تعمل بها أداة فحص الكائن Object Inspector.

المكون TComboBoxEx :

هذا المكون عبارة عن أداة تحكم لقائمة منسدلة جديدة وهو يسمح للصور بأن تظهر بجوار العناصر الموجودة بالقائمة المنسدلة.

المكون TColorBox :

هذا المكون عبارة عن أداة تحكم لقائمة منسدلة جديدة تكون مخصصة للتعامل مع الألوان واختيارها.

المظاهر والإمكانيات التى تم تطويرها

أغلب أدوات التحكم التى يتم التعامل معها من خلال بيئة الويندوز أصبحت الآن تعمل على نشر خصائص معالجة الحواف Bevel التالية :

- الخاصية BevelEdges.
- الخاصية BevelInner.
- الخاصية BevelOuter.
- الخاصية BevelKind.
- الخاصية BevelWidth.

المكونات الفرعية Subcomponents

أصبح في إمكان المكونات الآن أن تمتلك مكونات أخرى تعمل على إنشاء مكونات فرعية. فعلى سبيل المثال يمكن لأى مكون أن يكون لديه خاصية تعتبر فى حد ذاتها إشارة مرجعية لمكون وهذه الإشارة المرجعية يمكن أن تكون داخلية (لمكون فرعى) أو خارجية (لمكون عادى). هذا ولو أن الإشارة المرجعية داخلية فى هذه الحالة لا يكون المكون الفرعى مملوكا للفورمة نفسها ولكن يكون مملوكا للمكون الموجود فى الفورمة. وهذا يعنى أنه فى إمكان المكونات الآن أن تقوم بنشر خصائص المكون الفرعى بطريقة صحيحة. بالإضافة لما سبق نجد أن أداة فحص الكائن Object Inspector قد تم تعديلها لكى تسمح لك بأن تشاهد خصائص المكونات المرجعية مباشرة inline (مثل الخاصية Font على سبيل المثال).



لكى يتقوم بإنشاء مكون يكون مشتملا على مكون فرعى فإن ذلك يتطلب استدعاء TComponent.SetSubComponent.

خصائص الـInterface القابلة للنشر

أصبح من الممكن الآن نشر خصائص الـInterface (مجموعة من الخصائص تعمل على تحديد نوعية الـInterface) وذلك فى حالة واحدة فقط وهى أن تكون أداة تنفيذ الخاصية عبارة عن مكون streamable.

هذا يعنى أنك تستطيع الآن مشاهدة الخصائص التى تأخذ Interface فى أداة فحص الكائن Object Inspector مما يؤدى إلى توفير قائمة منسدلة تضم المكونات التى تعمل على تدعيم الـInterface.

الإضافات والتغييرات التى أجريت لوحدات البرمجة

فيما يلى سنتعرف معا على الإضافات والتغييرات التى أجريت لوحدات البرمجة فى الإصدار السادس للغة Delphi

الوحدة CheckLst.pas :

أصبح الآن فى إمكان المكون TCheckListBox نشر العديد من الخصائص الجديدة بما فيها الخصائص التالية :

- الخاصية AutoComplete.

- الخاصية HeaderColor.
- الخاصية HeaderBackgroundColor.

الوحدة **Classes.pas** :

- المكون TList أصبح الآن يمتلك طريقة تخصيص Assign جديدة لا تعمل فقط على إجراء عملية النسخ ولكنها أيضا تسمح بإجراء مجموعة من العمليات الأساسية.
- المكون TCollection أصبح الآن يمتلك إثناء من الطرق المحمية التى يتم استخدامهما للسماح بعمل سلالات للمكون TCollection وذلك بهدف تحسين وتطوير عملية إضافة ومسح العناصر. وهذين الأسلوبين الجديدين هما الأسلوب Added والأسلوب Deleting. ونود هنا القول بأنه لا يتم تنفيذ كلا الأسلوبين على الفور أى أنهما لا يمتلكان خاصية التنفيذ الإفتراضى default implementation. وعلى العموم فليس هناك كل من الحدث OnAdded والحدث OnDeleting لجعل الكائن TCollection صغيرا. ولكن القطاعات المتشعبة من هذا الكان يمكنها بسهولة إضافة هذه الأحداث. بالإضافة إلى ما سبق نقول أن الكائن TCollection لديه خاصية Owner جديدة لجعل من السهولة بمكان تعريف وتحديد المالك للكائن TCollection.
- المكون TStringList أصبح لديه الآن الخاصية الجديدة CaseSensitive والتى تسمح لك بأن تتحكم فى تحديد عمليات عرض السلاسل الحرفية (الفرز والترتيب Sorting والتطابق بين السلاسل الحرفية) التى تتأثر بحالة الحروف (صغير أم كبيرة).
- لقد تم نقل المكون TDataModule هنا (Delphi 6) من وحدة الفورم Forms وذلك لإزالة الإعتماديات على أدوات التحكم المرئية. وهذا يسمح لك بأن تكتب تطبيقات خادم أصغر حجما حيث إنها لا تتضمن أى واجهة استخدام للتفاعل مع المستخدمين.

- المكون TThread أصبح لديه الآن الخاصية الجديدة FatalException التى تعمل على تعريف وتحديد لأى استثناء يمكن أن يؤدى إلى إيقاف دالة الـThread من أن تكتمل بشكل طبيعي.
- المكون TStream وهو يعمل على إجراء تحميل زائد أو فوقى للدالة SEEK وذلك للسماح للقيم Int64 بأن يتم استخدامها لتعريف وتحديد المواضع. أما بالنسبة للقطاعات المرتبطة بهذا المكون فيمكن أن تتخطى واحد أو التحميل الفوقى الآخر ولكن لا ينبغى أن تتخطى كلاهما.
- المكون TInterfacedPersistent وهو عبارة عن قطع تأسيس جديد للكائنات المتبقية دوما والتى لا تعتبر مكونات ولكن تتولى مهمة تنفيذ الـInterfaces.

الوحدة ComCtrls.pas :

بهذه الوحدة تم إضافة المكون TTReeView الذى يتولى مهمة إنشاء نقط التواصل داخل الهيكل الشجرى والتى تعرف بـTreeNodes.

- لقد تم الآن تعميم عملية إنشاء نقط التواصل nodes وإعداد حدث ومن ثم نجد أن مستخدمى الهياكل الشجرية البسيطة لن يحتاجوا بعد الآن إنشاء تفرعات وتشعبات بالهيكل الشجرية وذلك لتخطى قطاع نقطة الاتصال. كذلك فقد تم تغيير AddNode ومن ثم يمكنم الآن المرور عبر نقطة الاتصال (مهما كان القطاع) التى تود إضافتها للهيكل الشجرى.
- لقد تم إضافة الحدث OnAddition والذى يقع عند إضافة نقط اتصال للهيكل الشجرى.
- لقد تم تبسيط وتنقيح مجموعة الدوال API الخاصة بعملية فرز وترتيب محتوى المشاهد الشجرية TreeViews. ومن ثم تستطيع الآن فرز وترتيب الهياكل الشجرية الفرعية بالإضافة إلى إمكانية ترتيب نقط الاتصال الموجودة فى قمة الهيكل الشجرى. ونود هنا القول بأن كل من القطاع TCustomTreeView والقطاع TTreeNode أصبحا الآن يقدمان تعريفات أكثر تجانس لكل من الأسلوب

AlphaSort والأسـلوب CustomSort عـلما بأن هذه الأساليب قد تم إضافتها إلى TTreeNodes.

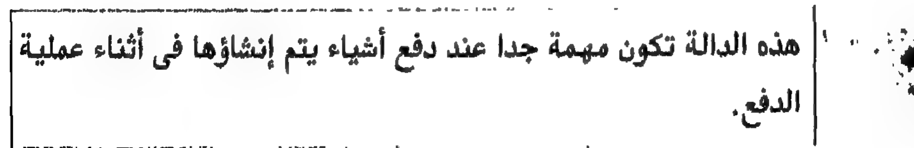


كافة التغييرات السالفة الذكر لم تؤثر على توافق هذه المكونات مع الإصدارات السابقة للغة Delphi.

- لقـد تم إضـافة كـائن الاختـيار المـتعدد MultiSelect وهـو يتمتع بأربعة خصائص وثمانية أساليب.
- لقـد تم إضـافة الخاصـية CreateListItems إلى المكـون TListView وهى تتطابق مع الخاصية CreateTreeNodes للمكون TTreeView.
- الآن تظهر نقط تغيير الحجم حول المكون TStatusBar حتى ولو كان شريط الحالة غـير مرتـبط مباشـرة بالفورمـة الموجـود بهـا. وطالما أن الركن الأيمن السفلى لشريط الحالـة واقعـا عـند الركن الأيمن السفلى للفورمة فى هذه الحالة ستظهر نقط تغيير الحجم على شريط الأدوات.
- لقـد أصبح لـدى الكـائن TDateTimePicker الخاصية الجديـدة Format وهى تـتحكم فـى طـريقة تنسـيق القـيم التاريخـية وذلـك باسـتخدام السلاسـل الحرفية القياسية الخاصة بتنسيق التاريخ أو الوقت.
- لقـد أصـبح لـدى THeaderControl عددا من الخصائص والأحداث الجديدة لكى يـتمكن مـن تدعـيم عملـية سـحب وإسـقاط الأعمـدة. ونود هنا القول بأن الخاصية الجديـدة HotTrack تسـمح بأن يتم التعليم على أقسام الرأس Header وذلك عند تعامل المستخدم مع هذه الأجزاء من خلال الماوس.
- لقـد أصبح لدى TToolBar الخاصية Menu الجديدة والتى تعمل على ملء شريط الأدوات بالأيقونـات المـناظرة للعناصـر الموجـودة بنظام القوائم. ونود هنا القول بأن هـناك عدد من الأحداث الجديدة تقع عندما يقوم المستخدم بتفصيل شريط الأدوات باستخدام أى من صناديق حوار التفصيل.

الوحدة Contnrs.pas :

- لقد تم إضافة كل من Last وFirst إلى كل من TObjectList و TComponentList وTClassList والتى تعتبر دوال ذات أنواع خاصة أو مصنعة Typecast.
- لقد تحولت الخاصية Push لكل من TStack وTQueue وTObjectStack و TObjectQueue إلى دالة تعطى بكل بساطة العنصر الذى تم دفعه إلى الـStack. وعليك أن تفكر فيها على اساس أنها Push/Peek.



هذه الدالة تكون مهمة جدا عند دفع أشياء يتم إنشاؤها فى أثناء عملية الدفع.

- يمكن استخدام كل من TBucketList وTObjectBucketList لإنشاء جداول بسيطة.

الوحدة Controls.pas :

- القطاع TCustomListControl عبارة عن قطاع تأسيس جديد مشترك لأدوات التحكم التى تعمل على تمثيل قائمة تضم مجموعة من العناصر (مثل القوائم المنسدلة Combo Boxes وقوائم العرض List Boxes وقوائم المشاهدة List Views). وهذا القطاع يقدم عدد من الأساليب الجديدة للتعامل مع القوائم Lists التى ستتوارثها بعد ذلك كافة التشعبات المتفرعة من هذا القطاع.
- كل من TDragObjectEx وTDragControlObjectEx و TDragDockObjectEx عبارة عن ثلاثة كائنات سحب جديدة وهى تتحرر تلقائيا عند إنتهاء عمليات السحب. وهذه الكائنات تناظر لكل من TDragObject وTDragControlObject وTDragDockObject ولكن الإختلاف الوحيد بينهم يتمثل فى أن الكائنات القديمة كانت لا تتحرر عند إنتهاء عملية السحب.
- الكائن TControl أصبح لدية الأن أسلوبين جديدين وهما الأسلوب ClientToParent والأسلوب ParentToClient. وهذه الأساليب الجديدة تسمح لك بأن تعد خريطة لنقط الشاشة pixels بحيث يتم ربطهم بواحد من الـParents أو الـChildren. وهذه

الأساليب تعمل بطريقة مماثلة جدا لطريقة عمل كل من الأسلوب ClientToScreen والأسلوب ScreenToClient وهما من الأساليب القديمة.

- الكائن TWinControl أصبح لدية طريقة جديدة لعمل تحميل فوقى أو زائد للأسلوب PaintTo الذى يتعامل مع نسيج canvas بدلا من HDC.
- لقد تم نقل المكون TModalResult من الوحدة Forms.pas إلى الوحدة Controls.pas. بالإضافة لذلك فإن دوال التدعيم التالية قد تم إضافتها إلى الوحدة Controls.pas:

```
function IsPositiveResult(const AModalResult: TModalResult): Boolean;
function IsNegativeResult(const AModalResult: TModalResult): Boolean;
function IsAbortResult(const AModalResult: TModalResult): Boolean;
function IsAnAllResult(const AModalResult: TModalResult): Boolean;
function St ipAllFromResult(const AModalResult: TModalResult):
TModalResult;
```

الوحدة **DbCtrls.pas** :

- كل من الكائن TDBLookupListBox وTDBLookupComboBox أصبح لدية الخاصية الجديدة NullValueKey والتى تسمح للمستخدمين بأن يخصصوا قيم فارغة NULL لهذه الكائنات.
- أصبح الآن لدى الكائن TDBComboBox كل من الخاصية الجديدة AutoComplete والخاصية AutoDropDown.
- كذلك فإن الكائن TDBListBox أصبح لدية الخاصية الجديدة AutoComplete أيضا.
- الكائن TOpenDialog أصبح يمتلك الآن الخاصية OptionsEx التى تعمل على تطوير وتوسيع قدرتك للتحكم فى كل من صندوق حوار الفتح Open وصندوق حوار الحفظ Save.

الوحدة **ExtCtrls.pas** :

- لقد تم إضافة الخاصية Proportional إلى أداة التحكم TImage. وهذه الخاصية تبقى على النسبة بين طول وعرض الصورة ثابتة بغض النظر عن الحجم الحالى لأداة التحكم TImage.

الوحدة **Forms.pas** :

- كل من TApplication وTApplicationEvents أصبح لديهما الحدث الجديد OnSettingChange الذى يسمح لك بأن تستجيب للتغييرات التى تجرى على القيم التحديدية الأساسية لنظام التشغيل.
- لقد أصبح الآن TForm يعمل على تدعيم الفورم ذات الطبقات :Layered form وذلك من خلال مجموعة الخصائص الجديدة AlphaBlend و AlphaBlendValue وTransparentColor وTransparentColorValue.
- الكائن TScreen لدية مجموعة من الخصائص الجديدة يتم استخدامها للحصول على منطقة عمل بسطح المكتب (هذه الخصائص هى WorkAreaRect و WorkAreaTop وWorkAreaLeft وWorkAreaHeight وWorkAreaWidth). بالإضافة إلى أنه يمتلك مجموعة من الأساليب الجديدة التى تسمح لك بأن تحدد أفضل موضع لمنطقة العمل بسطح المكتب.
- يعمل الكائن TMonitor على تطوير وتقوية الدعم الخاصة بإمكانية التعامل مع عدة شاشات فى نفس الوقت وذلك عن طريق تحديد أى من الشاشات تعتبر الشاشة الأساسية أو الرئيسية. كما أن هذا الكائن يمتلك كل من الخاصية WorkAreaRect والخاصية BoundsRect.
- لقد تم إضافة الدعم AutoDragDocking لهذه الوحدة. وهذا الدعم يسمح لك بأن تغلق خاصية الرسو التلقائى auto-docking للتطبيق الذى تتولى إعداده. كذلك فإنه تم إضافة flag إلى صندوق حوار الخيارات Options الخاص بلغة Delphi مما يسمح لك بأن تحدد قيمة هذه الخاصية.
- لقد تم نقل TModalResult إلى الوحدة Controls.pas.

الوحدة Graphics.pas :

- لقد تم إضافة كل من TFontRecall وTPenRecall وTBrushRecall إلى هذه الوحدة. وهذه المكونات تسمح بأن يتم سريعا حفظ واستعادة الفونتات والأقلام pens والفرش. وهذه المكونات متشعبة من TRecall (التابع لقطاعات الفورم) الذى يعمل مع القطاعات TPersistent بصفة عامة.
- لقد تم تخزين ألوان النظام وذلك لجعل من السهولة بمكان العثور على الألوان المطلوبة.
- لقد تم إضافة الألوان الأربعة الجديدة التالية إلى الألوان الستة عشر القياسية :

- clMoneyGreen.
- clSkyBlue.
- clCream.
- clMedGray.

الوحدة ImgList.pas :

- لقد تم إضافة أدوات للتحميل الفوقى أو الزائد لكل من الأسلوب Draw والأسلوب DrawOverlay والأسلوب GetIcon مما يسمح لك بأن تتخطى القيم التحديدية للخصائص الخاصة بقائمة عرض الصور Image List.

الوحدة IniFiles.pas :

- الأن أصبحت الملفات التى لها الإمتداد ini. تعمل على تدعيم عمليات كتابة وقراءة بيانات ثنائية Binary وذلك باستخدام stream.
- الآن يسمح لك المكون TMemIniFile بأن تتحكم فيما إذا كانت السلاسل الحرفية سيتم معالجتها بطريقة تأخذ فى الإعتبار حالة الحروف (كبيرة أو صغيرة) أم لا.
- يعتبر المكون THashedStringList فرع جديد من القطاعات TStrings وهو يستخدم جدول داخلى بسيط وذلك بهدف تطوير وتحسين مستوى أداء التطبيق.

الوحدة Masks.pas :

- بهذه الوحدة نجد أن كل من EditMask وText يستخدمان الآن أنواع خاصة أو مفصلة ومن ثم أصبحت خصائص التحرير الخاص بهما أكثر إفادة.

الوحدة Menus.pas :

- المكون TMenuItem لدية الآن الخاصية الجديدة AutiCheck التى تحدد إمكانية مراجعة أو عدم مراجعة العناصر الموجودة بنظام القوائم تلقائيا وذلك عندما يتعامل المستخدم مع هذه العناصر بالتطبيق.

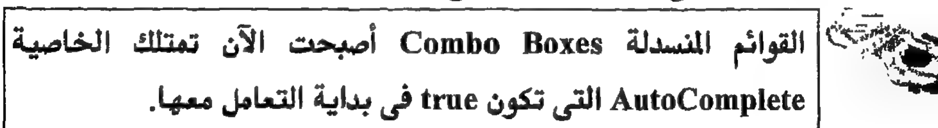
الوحدة Registry.pas :

- أصبح المكون TRegistry الآن يعمل على تدعيم كل من قراءة وكتابة البيانات الثنائية باستخدام Stream.

الوحدة StdCtrls.pas :



- لقد تم إضافة كل من الحدث OnCloseUp والحدث OnSelect إلى المكون TCustomComboBox (ومن ثم فقد تم إضافتهما أيضا إلى المكون TComboBox).
- تتم إستثارة الحدث OnCloseUp عند إغلاق القائمة المنسدلة الخاصة بالكائن Combo Box (عليك أن تفكر فى ذلك على أساس أنه معاكس للحدث OnDropDown).
- بالنسبة للحدث OnSelect فتتم استثارتة عند اختيار أى عنصر من القائمة المنسدلة للكائن Combo box (أو عندما يحدث تغيير لمحتوى القائمة المنسدلة عن طريق التجول فى هذا المحتوى لأعلى أو لأسفل).



القوائم المنسدلة Combo Boxes أصبحت الآن تمتلك الخاصية AutoComplete التى تكون true فى بداية التعامل معها.

- الآن يعمل TListBox على تدعيم نوعين جديدين من الأنماط وهما النمط IbVirtual والنمط IbVirtualOwnerDraw. وهذه الأنماط تكون مخصصة لقوائم العرض التصورية Virtual List boxes والتى لا تحتفظ بأى عناصر داخلها. ولكن بدلا من ذلك فإنك تشير إلى عدد العناصر الموجودة بهذه النوعية الخاصة من قوائم العرض عن طريق تخصيص قيمة للخاصية Count وبعد ذلك تقوم بإمدادهم بالعناصر (وبالكائنات المرتبطة بهم) وذلك باستخدام مجموعة الأحداث الجديدة OnData وOnDataFind وOnDataObject.

الوحدة TypInfo.pas :

- أصبح من الممكن الآن استدعاء الدالة GetPropInfo مع أى كائن ليس لدية أى معلومات RTTI. وفى مثل هذه الحالة فإننا نحصل من الدالة على nil.
- لقد تم إضافة المكون FreeAndNilProperties إلى هذه الوحدة. وهذا المكون سيأخذ أى كائن RTTI-enabled object ثم يقوم بتفريغ الخصائص الخاصة به مع ملاحظة أنه سيتم أيضا تفريغ أى كائنات مشار إليها مرجعيا من خلال هذا الكائن.

الوحدات RTL الجديدة والمظاهر الجديدة الخاصة بها (كافة الإصدارات)

بعض الدوال قد تم نقلها من وحدات أخرى إلى وحدة النظام فى حين أن العديد من دوال النظام قد تم نقلها هى الأخرى إلى وحدة المتغيرات الجديدة. بالإضافة إلى ذلك فقد تم أيضا إضافة العديد من إصدارات التحميل الفوقى الجديدة للدوال RTL وذلك من أجل تدعيم المعاملات WideString بالإضافة إلى المعاملات AnsiString (SysUnits) مثل المعامل Trim والمعامل WideFormat.

الوحدة Variants.pas :

- هذه الوحدة الجديدة تشتمل على العديد من الخدمات للتعامل مع المتغيرات. والكثير من هذا الكود البرمجى قد تم نقله من الوحدة System ومن ثم الآن لو أن الكود البرمجى الذى تعده يستخدم متغيرات فى هذه الحالة ينبغى عليك إضافة Variants إلى الجملة Uses. كذلك فإن هذه الوحدة تشتمل أيضا على عدد من الروتينات التى تعمل على تدعيم استخدام Variants وفى نفس الوقت تدعم أنواع المتغيرات المفصلة أو الخاصة Custom.
- لكى يتم تدعيم إمكانية إعداد تطبيقات تكون قادرة على العمل عبر العديد من أنظمة التشغيل المختلفة نجد أن الخدمات الموجودة بالوحدة Variants لم تعد تجرى استدعاءات مباشرة داخل دوال الويندوز API. ولكن بدلا من ذلك هناك وحدة جديدة تسمى VarUtils.pas تشتمل على روتينات منخفضة المستوى low-level تعمل على توفير نظام تأسيس طبيعى من أجل الروتينات الموجودة بالوحدة Variants.pas. وأنت تحتاج لأن تتوخى الحرص عند استخدام الكود التوليدى Generic code بالوحدة VarUtils.pas وذلك بسبب أنه لا يشبه تماما كود الويندوز كما أنك لو قمت بالخلط بين استدعاءات الويندوز والكود التوليدى فى هذه الحالة ستقع فى مشاكل أنت فى غنى عنها.

الوحدة ConvUtils.pas :

- هذه الوحدة عبارة عن مجموعة من الروتينات الخاصة بعملية التحويل بين وحدات القياس التى تم إضافتها.

الوحدة StdConvs.pas :

- هـذه الوحـدة عبارة عن مجموعة من المتغيرات العامة Global التى يتم استخدامها مع الروتينات الموجودة بالوحدة ConvUtils.pas.

الوحدة DateUtils.pas :

- هـذه الوحـدة عـبارة عـن مجموعة من دوال التاريخ والوقت تم إضافتها حديثا للغة Delphi 6.

الوحدة StrUtils.pas :

- هـذه الوحـدة الجديـدة تشـتمل عـلى دوال حرفـية بالإضـافة إلى الـدوال الموجـودة بالوحدة SysUtils.pas.

الوحدة FMTBCD.pas :

- هـذه الوحـدة الجديـدة تشـتمل عـلى خدمـات للـتعامل مـع القيم BCD (اختصار للمصطلح Binary-Coded Decimal).

الوحدة Math.pas :

- لقـد تم إضـافة Const إلى مجموعة المعاملات القابلة للامتداد من أجل الإسراع من إجراء العمليات الرياضية. كذلك هناك العديد من الثوابت والدوال بهذه الوحدة.

الوحدة System.pas:

- الوحـدة System لديهـا العديـد مـن الروتيـنات الجديـدة والتى أغلبها تعمل على تدعـيم إمكانـية إعـداد وبـرمجة تطبـيقات تكـون قادرة على العمل عبر العديد من أنظمـة التشـغيل المخـتلفة كمـا تدعـم أيضـا إمكانية التحويل بين الأنظمة المختلفة لتكويد الحرفيات المستخدمه بكل من بيئة الويندوز وLinux.



الروتينات التى تعمل على تدعيم المتغيرات Variants قد تم نقلها من الوحدة System.pas إلى الوحدة Variants الجديدة.

- لقد تم إضافة IInterface إلى الوحدة System.pas لكى يتم استخدامها مع الـ Interfaces التى ليست COM.

الوحدة SysUtils.pas:

- لقد تم تحديث الوحدة SysUtils بحيث تعمل على تدعيم مظاهر وإمكانيات الـ Cross-Platform الخاصة بـCLX. وهذا التحديث يتمثل فى أن بعض الخدمات التى كان يتم توفيرها فى السابق من خلال دوال الويندوز API الأساسية أصبح الآن يتم توفيرها من خلال الروتينات الموجودة بالوحدة SysUtils. ولكن على كل حال العديد من هذه الروتينات يظهر فى المناطق المخصصة للتعامل مع نظام التشغيل Linux من ملف الكود البرمجى الأصلى وكما لا يتم ترجمتها إلى النظام الثنائى الذى تعتمدة بيئة الويندوز.

تدعيم أنواع المتغيرات الخاصة (كافة الإصدارات)

تستطيع الآن تعريف أنواع بيانات خاصة وتخصيصها لأى متغيرات. وهذه الإمكانية تعمل على تقديم معامل تشغيل التحميل الفوقى فى أثناء تخصيص النوع الخاص إلى المتغير. هذا ولإنشاء نوع متغير جديد عليك أن تكون فى مكون منحدر من القطاع أو TCustomVariantType (أو واحد من الكائنات المنحدرة منه TInvokeableVariantType أو TPublishableVariantType) ثم تعلن عن حالة instantiate لنوع المتغير الجديد.

فيما يلى سنشاهد معا إثنين من الوحدات الجديدة التى تقدم لنا أمثلة للمتغيرات الخاصة:

- الوحدة VarCmplx تقوم بتنفيذ نوع متغير خاص للأرقام المركبة. ونود هنا القول بأن نوع المتغير يعمل على تدعيم المعالجة المباشرة باستخدام معاملات التشغيل للجمع والطرح والضرب والقسمة (ولكن ليست قسمة القيم العددية الصحيحة Integer) والـnegation. وفى هذا الصدد نقول إن نوع المتغير يعمل على تدعيم ٥ خصائص وهى : Real وImaginary وRadius وTheta وFixedTheta. كذلك يمكن تحوير نوع المتغير لتكوين أنواع من البيانات العددية الصحيحة Integer Types وأنواع بيانات عددية عشرية floating point وأنواع بيانات حرفية

string وأنـواع بيانات تاريخية وزمنية TDateTime وكذلك أنواع بيانات منطقية Boolean. بالإضـافة لذلـك نجـد أن الوحـدة VarCmplx تقـوم بتنفـيذ عـدد من الدوال العامة من أجل إجراء عمليات على المتغيرات المركبة complex.

- الوحدة VarConv تعمل على تنفيذ إمكانية إعداد أنواع متغيرات خاصة وذلك من أجـل القياسـات ووحـدات القـياس مثل المستخدمه فى الوحدة ConvUtils. ونود هـنا القـول بـأن نـوع المـتغير الخـاص Convert يعمل على تدعيم عمليات الجمع والطرح والضرب والقسمة التى تجرى بين نوع المتغير الخاص Convert والأرقام أو بين إثنين من نوع المتغير الخاص Convert (فيما عدا أنك لا تستطيع ضرب إثنين مـن أنـواع المـتغيرات الخاصـة Convert فـى بعضـهما الـبعض فـى حال أنهما لا يسـتخدمان نفـس الوحـدات). نـوع المـتغير Convert يقوم تلقائيا بضبط الوحدات عـندما تقـوم بـأداء العملـيات السـالفة الذكر. وأنت تستطيع تحوير cast المتغيرات الـتى مـن الـنوع Convert بحيـث تصـبح OleStr وString وDouble. كمـا إنهـا تعمـل أيضـا عـلى تدعـيم الخصـائص Value وType وTypeName وFamily و FamilyName بالإضـافة للخاصية As<Unit> وذلك للحصول على كل من قيمة عدديـة ونـوع وحـدة القـياس واسمهـا ومجموعـة وحدات القياس التى تنتمى إليها وحـدة القـياس المسـتخدمة حالـيا. بالإضـافة إلى الحصـول عـلى القيمة الناتجة من تحويلها إلى وحدة قياس أخرى فى نفس عائلة وحدات القياس.

إمكانية إعداد تطبيقات تعمل عبر مختلف نظم التشغيل (الإصدار الفنى Professional والإصدار المتكامل Enterprise)

تـأتى لغـة Delphi 6 وهـى مـزودة بقطـاع مكتـبى يعرف بـCLX (اختصار للمصطلح Borland Component Library for Cross-Platform). وهـذا القطـاع المكتـبى يشبه القطاع VCL الـذى يمكـن تشـغيله فـى كل من بيئة الويندوز وبيئة Linux. وفى هذا الصدد نقول إن الكائـنات الـتى تنـتمى للقطـاع CLX قد تم تسميتها بنفس أسماء الكائنات التى تنتمى للقطاع VCL كمـا إن كـلا الـنوعين مـن الكائـنات لديهما العديد من الخصائص والأساليب والأحداث المتشـابهة. وأنـت تستطيع استخدام القطاع المكتبى CLX مع Delphi 6 وذلك من أجل إعداد تطبيقات يمكن ترجمتها من خلال كل من بيئة الويندوز وبيئة Linux.

الاختلافات فى القطاع المكتبى CLX

هـناك إخـتلافات نتـيجة لـنظام التشـغيل والمظاهـر والإمكانـيات الـتى ترتـبط بالتكنولوجيات التى تكون مقصورة على بيئة الويندوز فقط :

- هـناك تشـابه إلى حـد كـبير بـين القطـاع CLX والقطـاع VCL بالرغم من أن أداة الـــتحكم TWidgetControl بالقطـــاع CLX تحـــل محـــل أداة الـــتحكم TWinControl بالقطـاع VCL. ومـن ثـم فـإن المكونـات الأخـرى مـثل المكـون TScrollingWidget يكون لها نفس الأسماء بكلا القطاعين.
- بيـئة الـتطوير المـتكاملة IDE تسـتخدم مجموعـة فرعـية أصـغر من الكائنات فى النافذة Object Repository (الموضحة فى الشكل التالى) وفى صفحات بالليتة المكونـــات :

شكل توضيحى :

- كافـة الأكـواد الـبرمجية المسئولة عـن تـأمين المـتغيرات والـتى كانـت فى الوحدة System أصـــبحت الآن فـــى الوحدتـــين الجديدتـــين : Variants.pas و VarUtils.pas.
- تسـتطيع أن تقـوم بـتعريف أنـواع بيانات خاصة مفصلة Custom للمتغيرات التى تـتعامل معهـا. وهذه الإمكانية تؤدى إلى التقديم لعامل تشغيل للتحميل الفوقى فى أثناء تخصيص نوع البيانات الخاص أو المفصل للمتغير. ونود هنا القول بأنه لإنشاء نـوع بـيانات جديـد لأى مـتغير عليك أن تعد تشعب جديد من القطاع ثم تستخدم TCustomVariant ثـم تعـد حالـة إبتدائـية instantiate لـنوع البـيانات الجديد الذى سيتم تخصيصه للمتغير.



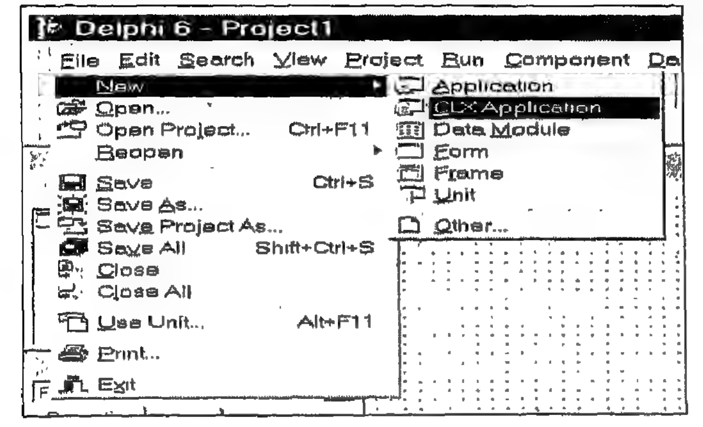
- يمكن استخدام أنماط التطبيقات المعدة للعمل عبر مختلف أنظمة التشغيل بالإضافة إلى خصائص الكائن OwnerDraw. وأنت تستطيع استخدام الخاصية Style للكائن TApplication وذلك لتوصيف وتحديد مظهر وسلوك العناصر الرسومية بهذه النوعية من التطبيقات.
- أغلب السلاسل الحرفية Strings الخاصة بأدوات التحكم الموجودة بالقطاع CLX تعتبر سلاسل حرفية متسعة Wide فى حين أن أغلب السلاسل الحرفية الخاصة بأدوات التحكم الموجودة بالقطاع VCL عبارة عن سلاسل حرفية من النوع ANSI. ونود هنا القول بأن إجراء عملية تحوير النوع Typecasting لسلسلة حرفية متسعة لتصبح PChar أو إجراؤها على النوع String ليصبح PWideChar لن تنجح وستتسبب فى جعل مترجم اللغة يعرض رسالة التحذير Suspicious typecast.
- نظام التشغيل Linux لا يستخدم أحقية استخدام registry لتخزين معلومات للتهيئة. ولكن بدلا من ذلك فإنك تستخدم ملفات التهيئة النصية و متغيرات بيئة العمل للقيام بهذه المهمة.

إنشاء تطبيق يعمل بمختلف أنظمة التشغيل cross-platform application

لكى تنشأ تطبيق يعمل بمختلف أنظمة التشغيل اتبع الخطوات التالية :

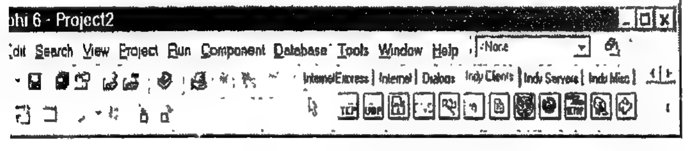
(١) افتح القائمة File ثم اختر منها New ثم CLX Application كما هو موضح بالشكل التالى :

شكل توضيحى :.



(٢) ستجد أن باليتة المكونات تتغير ديناميكيا لكى تعرض الكائنات التى تكون متاحة للاستخدام فى التطبيقات CLX التى ستعمل ببيئة الويندوز كما هو موضح بالشكل التالى :

شكل توضيحى :





هناك بعض المظاهر المعينة من مظاهر الويندوز لا يتم توفيرها للتطبيقات التى ستعمل بنظام التشغيل Linux.

أدوات الترجمة والمطورة (الإصدار المتكامل Enterprise)

مجموعة أدوات الترجمة Translation Tools والتى تضم كل من مدير الترجمة Translation Manager والمعالج Resource DLL Wizard بالإضافة إلى مستودع الترجمة Translation Repository قد أصبح اسمها بيئة الترجمة المتكاملة ITE (Integrated Translation Environment). والآن اصبح مدير الترجمة مستقلا بذاتة ويمكن تشغيله بصفة مستقلة مما يمكن استخدامه خارج بيئة التطوير المتكاملة IDE. ونود هنا القول بأنه عند استخدام مدير الترجمة من خارج بيئة التطوير المتكاملة IDE فى هذه الحالة يطلق عليه مدير الترجمة الخارجى ETM (External Translation Manager) (فى هذه الحالة يتم تشغيله من الملف etm60.exe) كما يمكن إرسالة إلى المترجمات بدون حاجتهم لأن يتم تركيب لغة Delphi.

الشكل التالى يوضح لنا صندوق حوار خيارات أدوات الترجمة Translation Tools Options

شكل توضيحى :

Translation Tools Options

Preferences | Colors | Fonts

Translation Repository
Filename :

Resource DLL Wizard
☑ Automatic repository query
☐ Automatically compile projects
☑ Show Translation Manager after RDW

Translation Manager
☑ Automatically quote strings

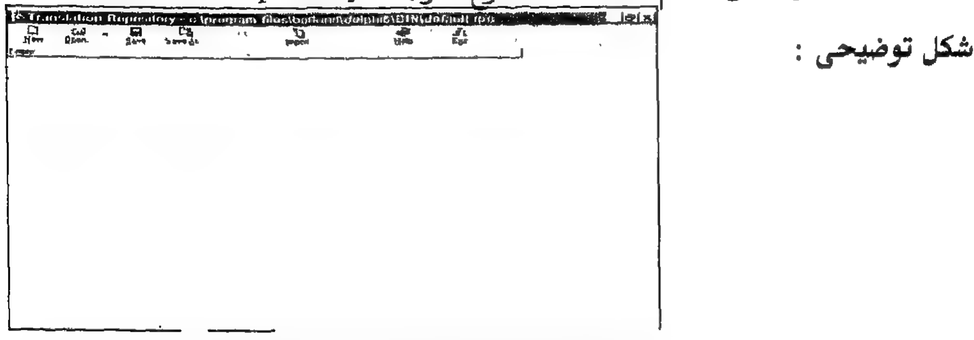
Miscellaneous
☐ Automatically save files

Multiple Find Action
○ Skip
● Use first
○ Display selection

OK | Cancel | Help

أما الشكل التالى فيقدم لنا نافذة مستودع الترجمة Translation Repository :

شكل توضيحى :



أما الشكل التالى فيوضح لنا نافذة مدير الترجمة الخارجى ETM :

شكل توضيحى :

صندوق الحوار الخاص بكل من مدير الترجمة TM ومدير الترجمة الخارجى ETM أصبحت تتمتع بالخصائص الوظيفية الجديدة التالية :

- أداة مشاهدة الفورمة التى منحت المستخدمين إمكانية تغيير حجم الفورمة فى أثناء إجراء عمليات الترجمة.
- تم تقسيم نافذة مدير الترجمة الخارجى ETM إلى ثلاثة تبويبات وهى تبويب بيئة العمل Environment وتبويب الملفات Files وتبويب منطقة العمل Workspace. وهذه التبويبات تساعد فى تنظيم المشروع الذى تتولى إعداده.
- تم إضافة كل من القائمة File والقائمة Project والقائمة Tools إلى نافذة مدير الترجمة الخارجى ETM كما تم إضافة شريط أدوات يضم أيقونات تناظر الأوامر الموجودة بهذه القوائم. وهذه الأوامر تعتمد بشدة على ما إذا كنت تقوم بتشغيل مدير الترجمة TM داخليا أم خارجيا.

التغييرات التى أجريت على طريقة نشر وتوزيع التطبيقات (كافة الإصدارات)

فـى الإصـدار الخـامس مـن لغـة Delphi كـان المسـتخدمين يقومـوا بنشـر الحـزم البرمجية التشغيلية runtime packages عن طريق توزيع الحزم المناسبة (الموجودة بالملفات ذات الامتداد BPL.) مع التطبيقات الخاصة بهم وذلك باستخدام InstallShield من أجل أن يتم تلقائيا ضم المكتبات الضرورية.

ولكـن الحـال تغير مـع الإصـدار السادس من لغة Delphi فهذا الأصدار يأتى وهو مـزود بالإصـدار الثالـث مـن InstallShield Express المبـنى عـلى تكنولوجـيا التركيـب Windows Installer (الـتى ابـتكرتها شـركة مايكروسـوفت) (وهـى تعـرف بـ MSI) كما يسـتخدم Modules الدمـج وذلـك لنشـر التطبـيقات التى يتم إعدادها. ونود هنا القول بأن Modules الدمـج عـبارة عـن مكونات MSI تشتمل على الملفات والمخطط المنطقى الضرورى لتركيب مكتبات مرحلة التشغيل Run-Time Libraries.



المكتبات المتاحة لدى لغة Delphi تمتلك الإعتمادات البينية مع Modules الدمج المتاحة أيضا لدى لغة Delphi والمصممة بحيث تتفاعل مع هذه المكتبات بكفاءه تامة.

زيادة كافة نظام المساعدة (كافة الإصدارات)

لغة Delphi الآن تسمح لك بأن تمرر طلبات المساعدة إلى أدوات مشاهدة محتويات نظام المساعدة التى تختلف عن الأدوات القياسية التى توفرها بيئة الويندور لمشاهدة مواضيع المسـاعدة ممـا يمكـنك مـن كتابة وإعداد أنظمة المساعدة للتطبيقات التى تتولى إعدادها سواء التى ستعمل ببيئة الويندوز أو بنظام التشغيل Linux.

فيما يلى سنستعرض سويا المظاهر والإمكانيات الجديدة التى أضيفت حديثا لنظام المساعدة :

- واجهـات الاسـتخدام الجديـدة New Interfaces والـتى تعـد وسيلة الاتصال بين التطبيق الذى تتولى إعداده وأدوات مشاهدة محتوى نظام المشاعدة. وهذه الواجهات الجديدة معرفة فى الملف HelpIntfs.pas وهى تضم كل من الآتى :
  - IExtendedViewer
  - ISpecialHelpViewer

- IHelpManager
- IHelpSystem
- IHelpSelector
- ICustomHelpViewer

- مدير المساعدة Help Manager والذى يحتفظ بقائمة تضم أدوات المشاهدة المسجلة والطلبات التى تم تمريرها إلى كل منهم.
- يعمـل القطـاع VCL عـلى توفـير إمكانية لتنفيذ ICustomHelpViewer المصممة للتحدث مع WinHelp.



مصممى ومطورى المكونات التى تنتمى للطائفة Third-Party لديهم الرغبة فى توفير نظام مساعدة باستخدام مولد المساعدة القياسى الذى تدعمه بيئة الويندوز. ولتحقيق هذه الأمنية يجب عليهم الآن توفير قائمة بكافة الـALinks المستخدمه فى ملف المساعدة المترجم Compiled. وهذه القائمة يجب أن تكون مخزنة فى ملف له الامتداد ALS. كما يجب وضع نسخة من هذا الملف داخل الفهرس الموصف بواسطة مدخل التسجيل للغة Delphi. هذا ويمكن إنشاء القوائم التى تضم ALinks من خلال ملفات المساعدة المترجمة وذلك باستخدام الخاصية Report فى ورشة عمل المترجم الخاص بنظام المساعدة (والمركب داخل مجلد الأدوات بالفهرس Help الموجود بدوره فى المجلد الذى تم تركيب لغة Delphi به.

الفصل الثانى

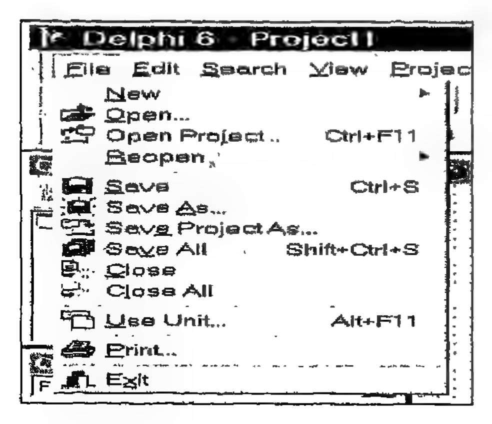
أدوات العمل
ببيئة التطوير المتكاملة IDE
للغة Delphi 6

القائمة File

يتم استخدام القائمة File لفتح وحفظ وغلق وطباعة المشاريع سواء كانت جديدة أو كانت موجودة بالفعل كما يمكن استخدامها أيضا لإضافة فورم ووحدات جديدة لفتح مشروع جديد.

الشكل التالى يوضح لنا محتويات هذه القائمة :

شكل توضيحى



الأمر New بالقائمة File

يعمل هذا الأمر على عرض القائمة الفرعية New كما يعمل أيضا على فتح صندوق الحوار New Items والذى يشتمل على الكائنات المخزنة فى مخزن الكائنات Object Repository كما يشتمل أيضا على مجموعة من المعالجات wizards المستخدمة فى إنشاء كائنات جديد.

الشكل التالى يوضح لنا محتويات القائمة الفرعية New :

شكل توضيحى :

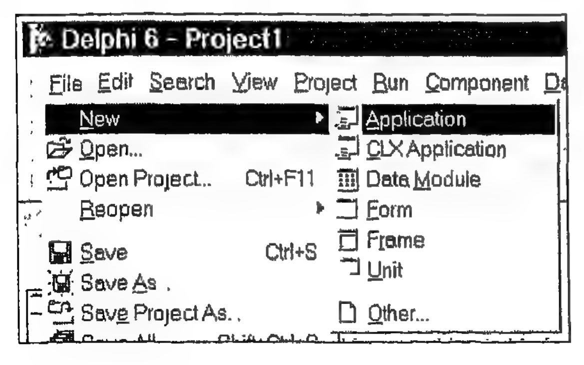
Delphi 6 - Project1
File Edit Search View Project Run Component
New — Application
Open... — CLX Application
Open Project... Ctrl+F11 — Data Module
Reopen — Form
Save Ctrl+S — Frame
Save As... — Unit
Save Project As... — Other...

الأمر Application بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء مشروع جديد يكون محتويا على فورمة فارغة ووحدة وملف مشروع.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

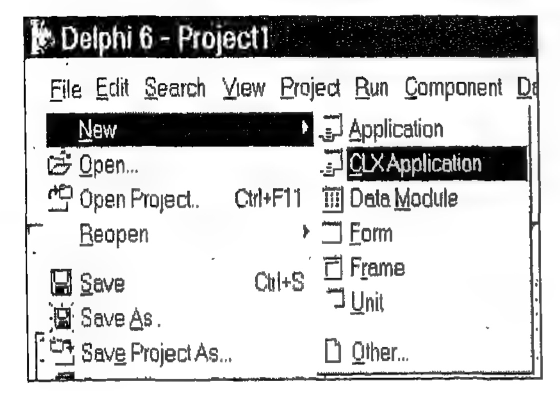


الأمر CLX Application بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء مشروع جديد محتويا على فورمة فارغة ووحدة وملف مشروع. وهذا المشروع يكون مخصص للعمل بنظام التشغيل Linux.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

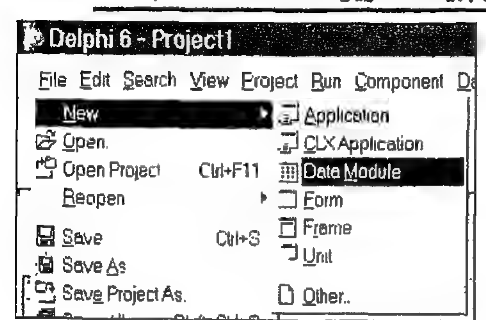


الأمر Data Module بالقائمة New بالقائمة File

من خلال هذا الأمر يتم إنشاء Module جديد للبيانات.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

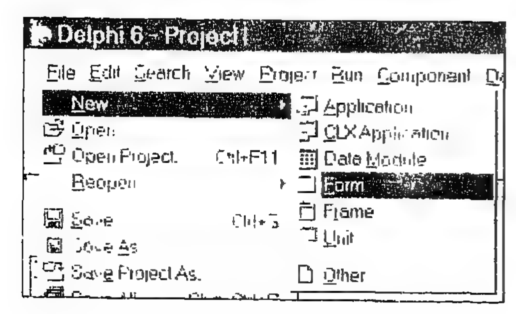


الأمر Form بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء فورمة فارغة كما يعمل على إضافتها إلى المشروع الحالى.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

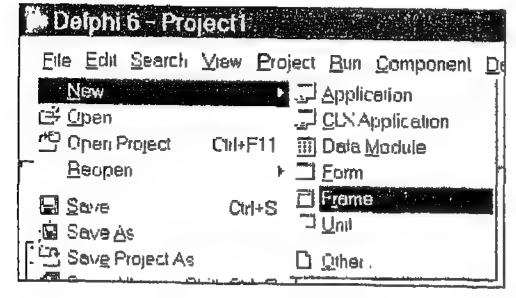


الأمر Frame بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء إطار فارغ وإضافته إلى المشروع الحالى.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

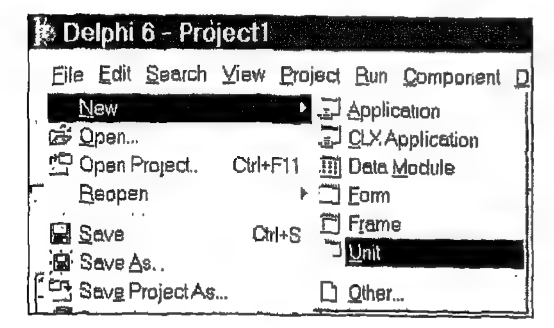


الأمر Unit بالقائمة New بالقائمة File

يعمل هذا الأمر على إنشاء وحدة جديدة بالمشروع الحالى.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :



الأمر Other بالقائمة New بالقائمة File

يعمل هذا الأمر على فتح صندوق الحوار New Items.

الشكل التالى يوضح لنا كيفية الوصول لهذا الأمر :

شكل توضيحى :

صندوق الحوار New Items

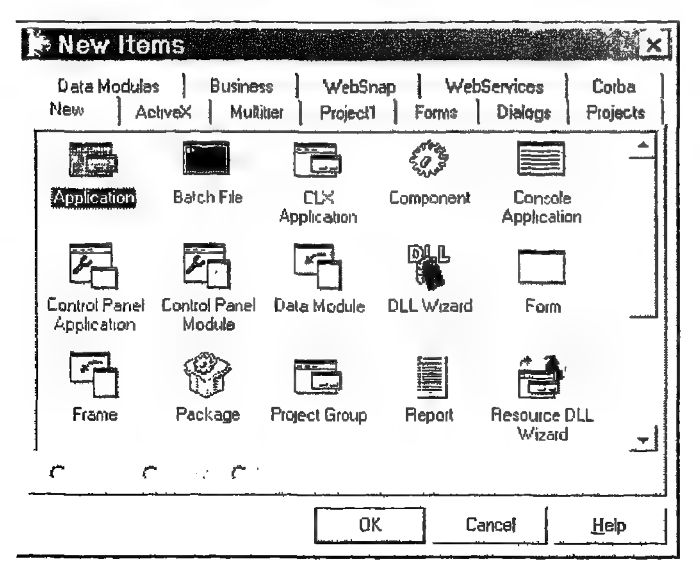
يعمل صندوق الحوار New Items على توفير وسيلة للوصول للعديد من الصفحات الموجودة بمخزن الكائنات Object Repository الذى يشتمل على صفحات مثل الصفحة Forms والصفحة Dialogs والصفحة Business التى تشتمل على تصنيفات وأقسام لعدد كبير ومتنوع من القوالب والكائنات والمعالجات التى يمكن استخدامها فى التطبيقات التى تتولى إعدادها من خلال لغة Delphi 6.

الصفحة New بصندوق الحوار New Items

الصفحة New بصندوق الحوار New Items تشتمل على العديد من العناصر المعدة مسبقا والتى يمكنك استخدامها فى أثناء إعداد التطبيق من خلال لغة Delphi.

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



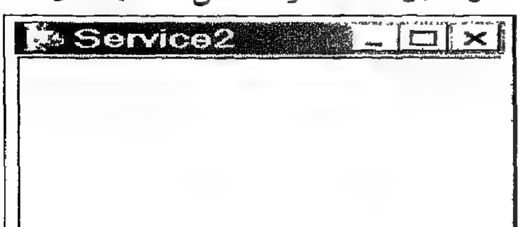
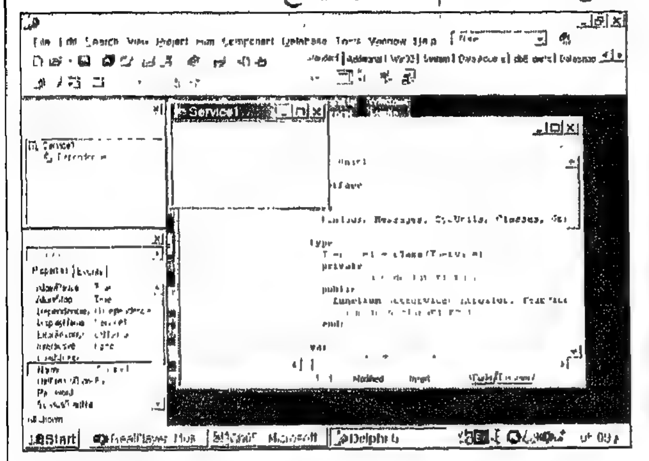
الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
Application	يقوم بإنشاء مشروع جديد يحتوى على فورمة ووحدة وملف له الامتداد DPR. كما إنه يوفر لك وسيلة لاختيار قالب template.
Batch File	يقوم بإنشاء مشروع ملف حزمة جديد Batch File يكون له الامتداد BAT. وهو يسمح لك بأن تحدد حزمة من الأوامر. ومثل هذه النوعية من المشاريع لا تكون مشتملة على أى فورم كما إنها لا تشتمل على محرر الكود البرمجى Code Editor.
CLX Application	يقوم بإنشاء مشروع لدية القدرة على العمل بأنظمة التشغيل المختلفة مثل Linux ومثل هذه النوعية من المشاريع تكون مشتملة على فورمة ووحدة وDPR.
Component	يقوم بإنشاء مكون جديد باستخدام المعالج Component Wizard الموضح فى الشكل التالى :

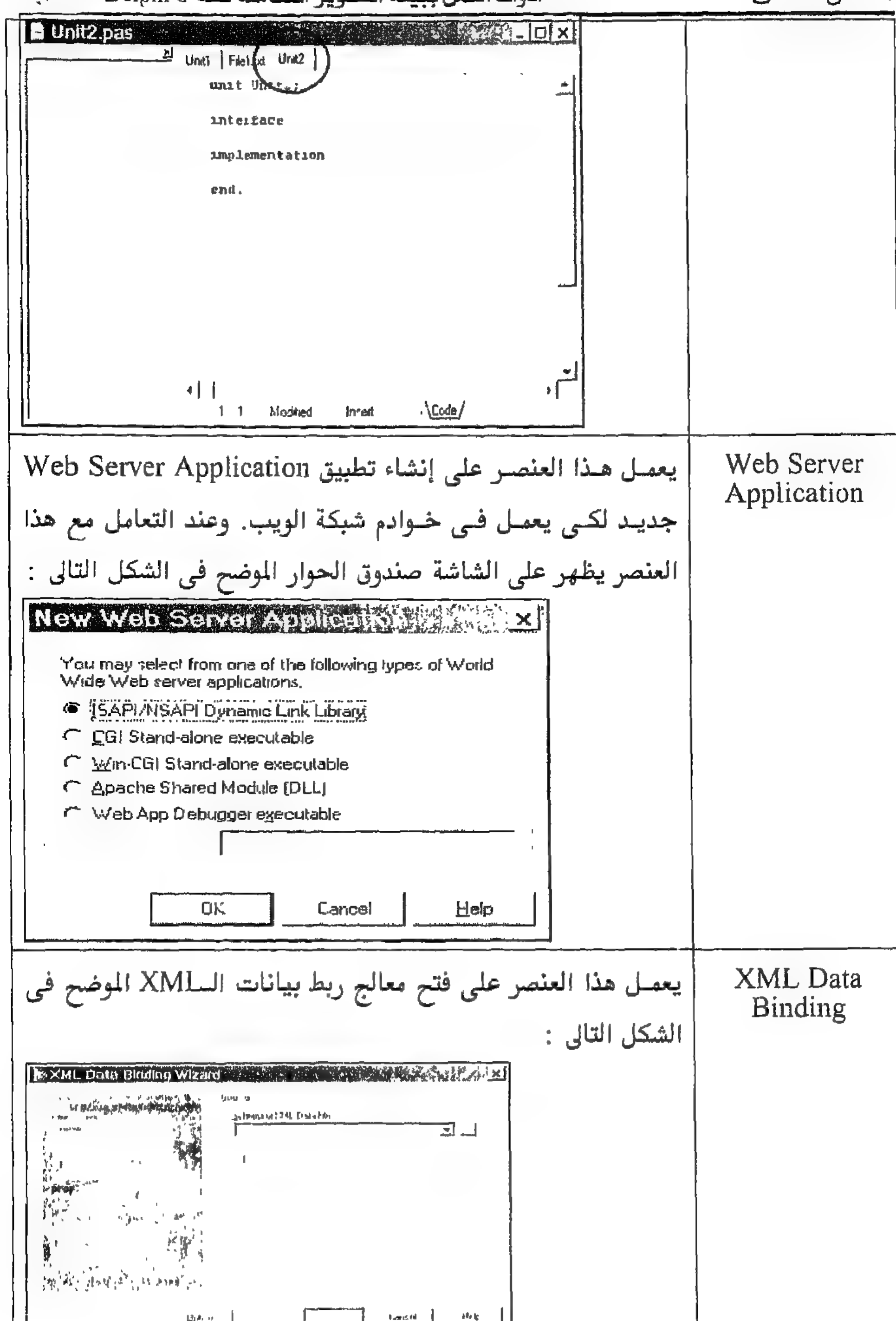
	New Component New Component Ancestor type: Class Name: Palette Page: Samples Unit file name: Search path: $(DELPHI)\Lib;$(DELPHI)\Bin;$(DELPHI)\Imports Cancel Help
Console Application	يقوم بإنشاء مشروع تطبيق جديد يعمل من خلال نظام التشغيل DOS ولا يحتاج لأن يعمل فى بيئة الويندوز.
Control Panel Application	يقوم بإنشاء applet جديد يمكن وضعه فى لوحة التحكم Control Panel الخاصة ببيئة الويندوز.
Control Panel Module	يقوم بإنشاء Module جديد لأى مشروع من المشاريع المصممة للعمل بلوحة التحكم ببيئة الويندوز.
Data Module	يقوم بفتح Module بيانات جديد فى أداة التصميم Data Module Designer الموضحة فى الشكل التالى : DataModule كما يقوم أيضا بعرض ملف الوحدة الخاص بالـModule الجديد وذلك فى محرر الكود البرمجى Code Editor كما يعمل على إضافة هذا الـModule إلى المشروع الحالى كما هو موضح بالشكل التالى :

DLL	يقوم بإنشاء مشروع DLL (اختصار للمصطلح Dynamic Link Library) والذى نحصل منه فى النهاية على ملف له امتداد DLL.
Form	يقوم بإنشاء وإضافة فورمة فارغة إلى المشروع الحالى كما يسمح لك أيضا بأن تختار قالب فورمة.
Frame	يقوم بإنشاء إطار جديد كالموضح فى الشكل التالى :
Package	يقوم بإنشاء حزمة برمجية Package. وفى هذا الصدد نقول إن الحزمة البرمجية الجديدة تظهر فى محرر الحزمة البرمجية Package Editor الموضح فى الشكل التالى :

Project Group	يقوم بإنشاء مجموعة مشروع Project Group جديدة لكى تشتمل على المشاريع المرتبطة معا. هذا وعن طريق إضافة المشاريع المرتبطة معا إلى مجموعة مشروع يمكنك من بناء كافة المشاريع من خلال أمر واحد. ونود هنا القول بأن ملف مجموعة المشروع يكون له الامتداد BPG.
Report	يقوم بإنشاء تقرير سريع يساعدك فى إنشاء تقارير قوية وفعالة لتطبيقات قواعد البيانات من خلال إمكانية التصميم على الشاشة بالنافذة Quick Report الموضحة فى الشكل التالى :
Resource DLL Wizard	يعمل على البدء فى تشغيل معالج Wizard لكى يساعدك فى تشغيل ملفات مكتبات الربط الديناميكية DLLs التى تشتمل على إصدارات محلية من فورم التطبيقات التى تعدها. هذا والشكل التالى يوضح لنا هذا المعالج :

Service	يعمل على إضافة خدمة جديدة إلى تطبيق من تطبيقات خدمات الـ NT التى سبق إنشاؤها. وهذا العنصر لا يعمل على إضافة خدمات إلى تطبيق لا يكون من تطبيقات الخدمات فى حين أن يمكن إضافة الكائن TService إلا إن التطبيث لن يقوم بتكوين الأحداث الخاصة بهذا الكائن كما لن يعمل الاستدعاءات المناسبة لبيئة الويندوز الخاصة بالخدمة. ونحن ننصحك هنا بأن لا تضيف أكثر من خدمة واحدة إلى أى تطبيق يكون فى حاجة إلى اسم المستخدم وكلمة السر لكى يبدأ فى العمل. هذا وعند النقر بالماوس نقرا مزدوجا على هذا العنصر تظهر النافذة الموضحة فى الشكل التالى : 
Service Application	هذا العنصر يعمل على إنشاء تطبيق جدمة NT Service Application جديد. فبمجرد أن تقوم بإنشاء تطبيق خدمة فسوف تشاهد نافذة فى أداة التصميم كما هو موضح بالشكل التالى :  وهذه النافذة تناظر خدمة (الكائن TService). هذا ويتم التعامل مع

	هذه الخدمة عن طريق تحديد قيم الخصائص الخاصة بها وذلك من خلال النافذة Object Inspector بالإضافة لإعداد الأكواد البرمجية لأدوات معاملة الأحداث المرتبطة به.
Text	يعمل هذا العنصر على إنشاء ملف نصى جديد من النوع ASCII وذلك من خلال المحرر الموضح فى الشكل التالى : C:\Program Files\Borland\Delphi6\Projects\File1.txt Unit1 File1.txt Insert Code
Thread Object	يعمل هذا العنصر على إنشاء كائن Thread جديد. وعند التعامل معه يظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى : New Thread Object Class Name: OK Cancel والذى يسأل عن اسم القطاع الذى سينتمى إليه الكائن Thread الجديد.
Unit	يعمل هذا العنصر على إنشاء وإضافة وحدة جديدة إلى المشروع الحالى. وهذه الوحدة الجديدة تظهر كتبويب جديد فى محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :



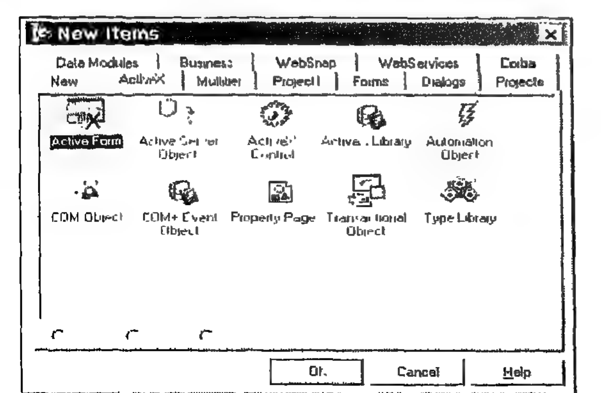
	Unit2.pas Unit1 | File1.txt | Unit2 unit Unit2; interface implementation end. 1 : 1 Modified Insert \Code/
Web Server Application	يعمـل هـذا العنصـر على إنشاء تطبيق Web Server Application جديـد لكـى يعمـل فـى خـوادم شبكة الويب. وعند التعامل مع هذا العنصر يظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى : New Web Server Application You may select from one of the following types of World Wide Web server applications. ISAPI/NSAPI Dynamic Link Library CGI Stand-alone executable Win-CGI Stand-alone executable Apache Shared Module (DLL) Web App Debugger executable OK Cancel Help
XML Data Binding	يعمـل هذا العنصر على فتح معالج ربط بيانات الـXML الموضح فى الشكل التالى : XML Data Binding Wizard

	وهـذا المعـالج يعتـبر أداة من أجل تكوين قطاعات تعمل على تمثيل مستندات معينة معدة بلغة XML. ومثل هذه النوعية من المستندات تعمـل عـلى توفـير طريقة بسيطة لتخزين المعلومات فى صورة نصية ومن ثم يكون من السهولة بمكان البحث عنها والعثور عليها.

الصفحة ActiveX بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



الكائـنات الموجـودة فى هذه الصفحة يتم استخدامها لإنشاء كائنات COM جديدة وفـورم فعالـة Active Forms وأدوات تحكـم ActiveX وصـفحات خصـائص Property pages لأدوات الـتحكم ActiveX بالإضـافة إلى مكتـبات الأنـواع الخاصـة بأدوات التحكم ActiveX أو كائنات التفعيل الذاتى Automation Objects.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
Active Form	يقـوم هـذا العنصـر بإنشاء فورمة فعالة جديدة Active form والتى تعتـبر أداة تحكـم ActiveX بسـيطة (تنحدر من TActiveForm) مهيئ مسبقا للعمل فى أى من برامج تصفح الإنترنت وشبكة الويب. هـذا وعـند الـتعامل مـع هـذا العنصـر يظهـر عـلى الشاشـة المعـالج ActiveX Control Wizard الموضح فى الشكل التالى :

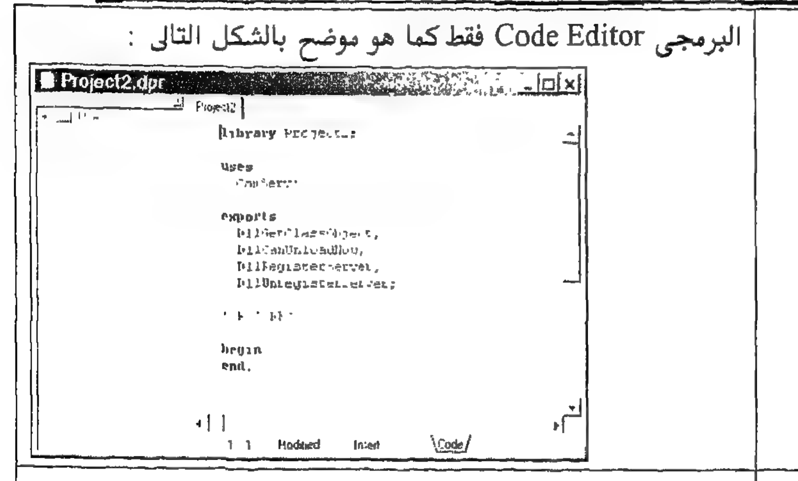
<table>
<tr>
<td></td>
<td>Active Form Wizard
New ActiveX Name: ActiveFormX
Implementation Unit: ActiveFormImpl1.pas
Project Name: ActiveFormProj1.dpr
Threading Model: Apartment
ActiveX Control Options
Make Control Licensed
Include About Box
Include Version Information
OK Cancel Help

وهو يرشدك في أثناء عملية الإنشاء مما يسمح لك بأن تضيف أدوات تحكم إلى الفورمة. وهذا المعالج يقوم بإنشاء كل من مشروع من النوع ActiveX Library (لو كانت هناك حاجة لذلك) ومكتبة أنواع وفورمة ووحدة تنفيذ بالإضافة إلى وحدة تشتمل على إعلانات مكتبة الأنواع المناظرة. هذا وينبغي عليك ملاحظة أنه بخلاف أدوات التحكم ActiveX الأخرى فإنك لا تستطيع من خلال بيئة التطوير المتكاملة IDE التعديل في قيم خصائص فورمة Active تم بناؤها إلا إذا أضفت كود برمجي يقوم بنشر هذه الخصائص.</td>
</tr>
<tr>
<td>Active Server Object</td>
<td>من خلال هذا العنصر يتم إنشاء ASP (اختصار للمصطلح Active Server Page) من أحد التطبيقات التي سبق إنشاؤها. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة صندوق الحوار Active Server Object الموضح في الشكل التالي :

New Active Server Object
CoClass Name:
Threading Model: Apartment
Instancing: Multiple Instance
Active Server Type
Page-level event methods (OnStartPage/OnEndPage)
Object Context
Options
Generate a template test script for this object
OK Cancel Help

الذى من خلاله تستطيع تحديد اسم CoClass ونموذج الـ Threading وهكذا...</td>
</tr>
</table>

ActiveX Control	مـن خـلال هـذا العنصـر يـتم إنشاء أداة تحكم ActiveX Control جديـدة. وعـند الـتعامل مـع هـذا العنصر يظهر على الشاشة المعالج ActiveX Control Wizard الموضح فى الشكل التالى : ActiveX Control Wizard VCL Class Name: New ActiveX Name Implementation Unit Project Name: Threading Model: Apartment ActiveX Control Options Make Control Licensed Include About Box Include Version Information OK Cancel Help هـذا المعـالج يرشـدك عبر عملية الإنشاء واختيار الكائن VCL الذى ترغـب فـى جعله الأساس لأداة التحكم الجديدة. هذا وينبغى عليك ملاحظة أن أدوات التحكم ActiveX تحتاج لمكتبة ActiveX لكى يتمكـنوا مـن امـتلاك واجهـات الـتعامل Interfaces الخاصـة بهـم وكذلـك المعـاملات الـتى يـتم تمريـرها إلى الأسـاليب الخاصـة بهـم وجعـلهم يـتعاملوا بفاعلـية مـع تطبـيقات العمـلاء Client Applications. هـذا ولـو لم يتم فتح مشروع مكتبة ActiveX قبل الشـروع فـى إنشـاء أداة تحكـم ActiveX فـى هذه الحالة تقوم لغة Delphi بفتح مشروع جديد من هذا النوع.
ActiveX Library	مـن خـلال هـذا العنصـر يمكن إنشاء مكتبة ActiveX جديدة. وفى الـبداية يـتم إنشـاء ملف قالب يسمى Project1.dpr وهو يعد نقطة الإنطـلاق بالنسـبة لك للمضى قدما بعد ذلك فى عملية الإنشاء. هذا وفـى حالـة عـدم فتح مشروع ActiveX Library قبل أن تشرع فى إنشـاء أداة تحكم ActiveX فى هذه الحالة تقوم لغة Delphi بفتح مشـروع جديد من هذه النوعية الذى يكون مشتملا على محرر الكود

	البرمجى Code Editor فقط كما هو موضح بالشكل التالى : 
Automation Object	يعمل هذا العنصر على إنشاء كائن تفعيل ذاتى Automation جديد. وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج Automation Object Wizard الموضح فى الشكل التالى : Automation Object Wizard CoClass Name Instancing: Multiple Instance Threading Model: Apartment Options Generate Event support code OK Cancel Help هذا المعالج يسمح لك بأن تدخل اسم قطاع للكائن Automation الجديد بالإضافة إلى إمكانية تحديد نوعية حالة الكائن وكذلك نموذج الـThreading. هذا وبمجرد أن يقوم المعالج بإنشاء الكائن Automation فإنه يتم استخدام محرر مكتبة النوع Type Library الموضح فى الشكل التالى :

	وذلك لتعريف واجهة الاستخدام التى سيستخدمها الكائن للعمل فى تطبيقات العملاء.
COM Object	من خلال هذا العنصر يمكن إنشاء كائن COM جديد الذى يستخدم فى إعداد واجهة استخدام مرحصة أو إنشاء واجهة استخدام جديدة تنحدر من IUknown. هذا وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج COM Object Wizard الموضح فى الشكل التالى : هذا المعالج يسمح لك بأن تحدد خصائص الخادم COM الجديد.
COM+ Event Object	من خلال هذا العنصر يتم إنشاء كائن COM+ Event جديد الذى يعمل على دفع أحداث الخادم لكافة العملاء المسجلة والمرخصة. وعند التعامل مع هذا العنصر يظهر على الشاشة المعالج COM+ Event Object Wizard الموضح فى الشكل التالى :

	هـذا المعالج يضيف كائن COM+ event إلى مشروع event object موجـود بالفعل أو يـبدأ فـى مشروع event object جديد من أجل هـذا الكـائن. وبعـد أن يقـوم المعـالج بـتكوين event object فإنـه يسـتخدم محـرر مكتـبة الـنوع لـتعريف واجهـة الاستخدام الخاصة بالكائن COM+ event كما هو موضح بالشكل التالى :
Property Page	مـن خـلال هـذا العنصـر يمكن إنشاء ملف يعمل على تهيئة وإعداد صفحة الخاصية لأداة تحكم ActiveX. وصفحة الخاصية تظهر فى مود التصميم وتكون جاهزة بحيث تمكنك من إضافة إعلانات خاصة وعامـة. وأنـت تسـتطيع تصـميم صندوق حوار فى نافذة الفورمة كما يمكـن تجمـيع الخصـائص فـى مجموعـة واحـدة وذلـك لجعـل مـن السـهولة بمكـان بالنسـبة للمبرمجين أن يقوموا بتعديل أداة التحكم عند التعامل معها فى أى تطبيق.

Transactional Object

مـن خلال هذا العنصر يمكن إنشاء كائن تفعيل ذاتى Automation يتمـتع بخدمات التطبيقات الموزعة التى يوفرها MTS أو +COM. هـذا وعـند الـتعامل مـع هـذا العنصـر يظهـر عـلى الشاشـة المعـالج Transactional Object Wizard الموضح فى الشكل التالى :

New Transactional Object

CoClass Name

Threading Model: Apartment

Transaction model
- Requires a transaction
- Requires a new transaction
- Supports transactions
- Does not support transactions
- Ignores Transactions

Options
- Generate Event support code

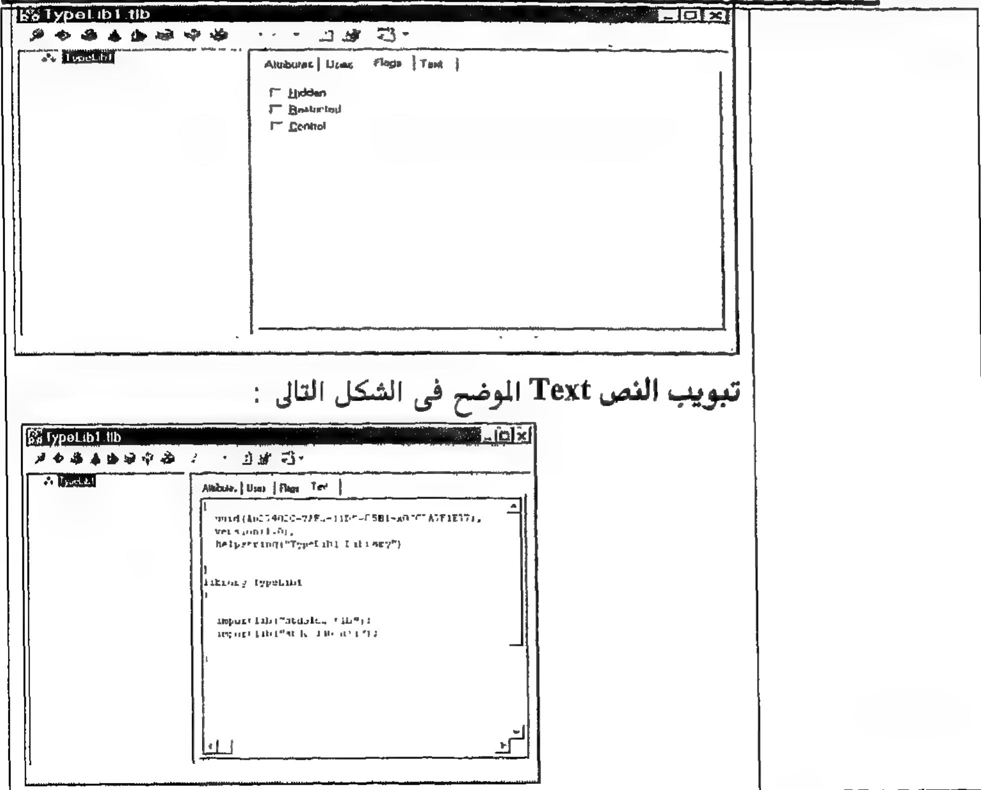
OK　Cancel　Help

هـذا المعـالج يسـمح لك بأن تحدد نموذج الحركة المطلوب. وبعد أن ينـتهى الـتعامل مـع المعـالج فإنـه يـتم اسـتخدام محرر مكتبة النوع لـتعريف واجهـة الاسـتخدام الـتى سيسـتخدمها الكائن للتعامل مع تطبـيقات العمـلاء. ونـود هـنا القول بأن لو كان من المفروض أن يتم تركيـب هذا الكائن تحت +COM فى هذه الحالة استخدم الصفحة +COM بمحرر مكتبة النوع الموضحة فى الشكل التالى :

Project1.tlb

وذلـك لـتحديد مجموعـة الاخـتـيارات الـتى سيتم اسـتخدامها عند تركيب الكائن داخل أى من التطبيقات +COM.

Type Library	مـن خـلال هذا العنصر يمكن إنشاء أو تحرير أى مكتبة من مكتبات الـنوع الـتى يمكـن استخدامها بواسطة مكتبات أنواع أخرى أو التى تعمـل عـلى تعـريف واجهـات الاسـتخدام الـتى يسـتخدمها المعالج COM Object Wizard لـتكوين الكائـنات. هـذا وعند التعامل مع هـذا العنصـر يظهـر عـلى الشاشة مباشرة محرر مكتبة النوع Type Library editor ولكـن هـنا نجـد أن المحـرر–ولـيس كمـا سـبق– يحتوى على أربعة تبويبات وهى : **تبويب الصفات Attributes** الموضح فى الشكل التالى : **تبويب المستخدمين Users** الموضح فى الشكل التالى : **تبويب الإشارات Flags** الموضح فى الشكل التالى :

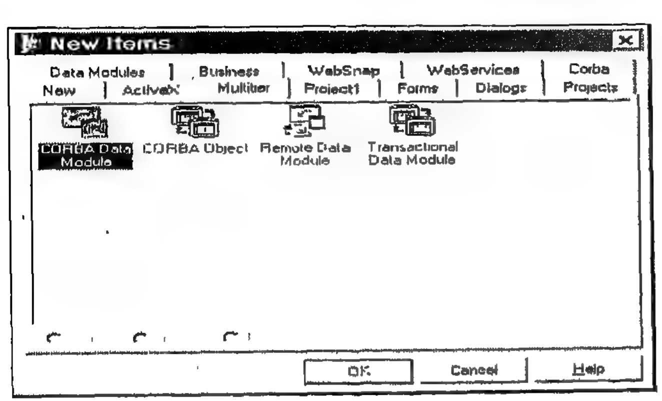


تبويب النص Text الموضح فى الشكل التالى :

الصفحة Multitier بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

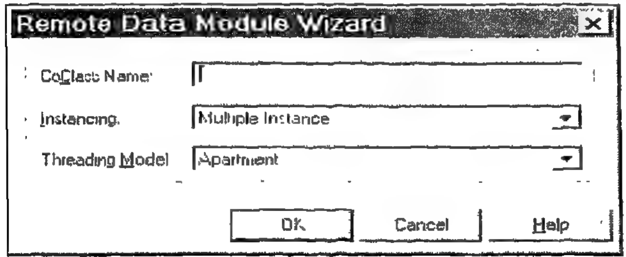
شكل توضيحى :



الكائنات والعناصر الموجودة فى هذه الصفحة يمكن استخدامها لإنشاء خوادم Servers تعتبر جزء من التطبيقات المتعددة الطبقات.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
CORBA Data Module	من خلال هذا العنصر يتم إنشاء CORBA Data Module لكى يعمل على أساس كونه خادم فى تطبيق قواعد بيانات متعدد الطبقات multi-tiered يستخدم CORBA على أساس كونها بروتوكولات إتصال. هذا ويتم التعامل مع هذا العنصر من خلال المعالج CORBA Data Module Wizard الموضح فى الشكل التالى : CORBA Data Module Wizard Class Name: Instancing: Instance-per-client Threading Model: Single-threaded OK Cancel Help هذا المعالج يقوم بتكوين التنفيذ الإبتدائى وتعريف قطاع تنفيذ لـ Module البيانات مع بعض الخيارات الخاصة بكل من عملية threading وعملية instancing.
CORBA Object	يعمل هذا العنصر على إنشاء كائن CORBA Server. هذا ويتم التعامل مع هذا العنصر من خلال المعالج CORBA Object Wizard الموضح فى الشكل التالى : CORBA Object Wizard Class Name: Instancing: Instance-per-client Threading Model: Single-threaded OK Cancel Help

<table>
<tr>
<td>من خلال هذا العنصر يتم إنشاء تطبيق Server فى تطبيق قاعدة بيانات متعدد الطبقات قائم على التكنولوجيا COM. هذا ويتم التعامل مع هذا العنصر من خلال المعالج Remote Data Module Wizard الموضح فى الشكل التالى :


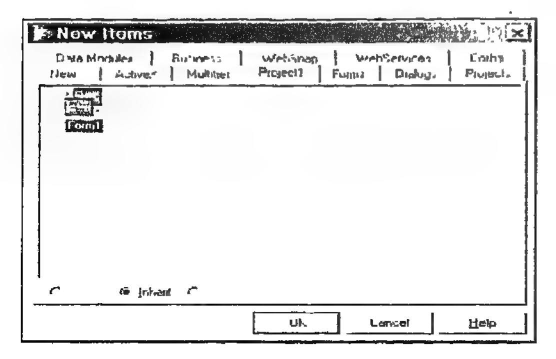
وهذا المعالج يعمل على تكوين التنفيذ الإبتدائى وتعريف قطاع التنفيذ لـ Module البيانات مع خيارات خاصة بكل من العملية Instancing والعملية Threading.</td>
<td>Remote Data Module</td>
</tr>
<tr>
<td>من خلال هذا العنصر يتم إنشاء Module للبيانات التى توجد فى مصادر بعيدة Remote Data والذى يمكن أن يستفيد من الخدمات التى يتم توفيرها من خلال MTS أو COM+. هذا ويتم التعامل مع هذا العنصر من خلال المعالج Transactional Data Module Wizard الموضح فى الشكل التالى :
New Transactional Data Module Object
CoClass Name:
Threading Model: Apartment
Transaction model:
Requires a transaction
Requires a new transaction
Supports transactions
Does not support transactions
Ignores Transactions
OK
Cancel
Help</td>
<td>Transactional Data Module</td>
</tr>
</table>

	وهـذا المعـالج يعمـل عـلى تكويـن التنفيذ الإبـتدائى وتعـريف قطاع التنفـيذ لــ Module البـيانات مـع خيارات خاصة بكل من العملية Instancing والعملية Threading.

الصفحة Project بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :


شكل توضيحى :

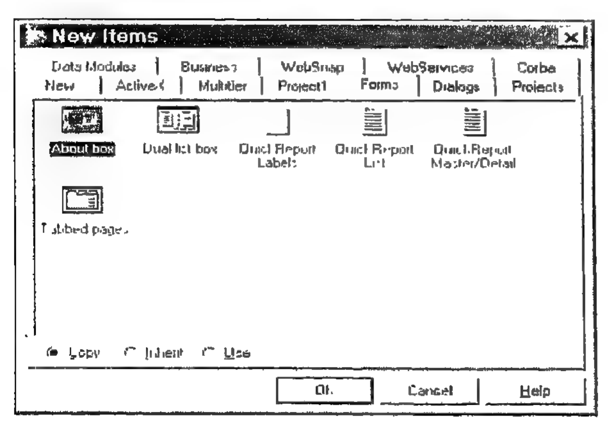


لـو أن لديـك مشـروع مفـتوح حاليا ستجد أن الصفحة Project فى صندوق الحوار New Items قـد أصبح اسمها هو نفس اسم المشروع المفتاح حاليا. كذلك تجد أيضا أن هذه الصـفحة تضـم كافة الفورم الموجودة بهذا المشروع. وأنت تستطيع إنشاء فورمة وريثة من أى فورمة فى أى مشروع سبق إعداده.

الصفحة Forms بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



مـن خـلال العناصر الموجـودة فـى الصـفحة Forms يمكن إنشاء أنواع مختلفة من الفورم فى المشروع الذى يتم التعامل معه حاليا.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
About Box	مـن خلال هذا العنصر يتم إنشاء صندوق حوار About الذى يعرض معلومـات عـن التطبيـق مـثل الاسـم ورقم الإصدار Version وتاريخ التصميم...والشكل التالى يوضح لنا صندوق الحوار هذا :
Dual List Box	مـن خـلال هذا العنصر يتم إنشاء فورمة تشتمل على إثنين من قوائم العـرض والـذى يمكـن تـبادل العناصر بينهما كما هو موضح بالشكل التالى :

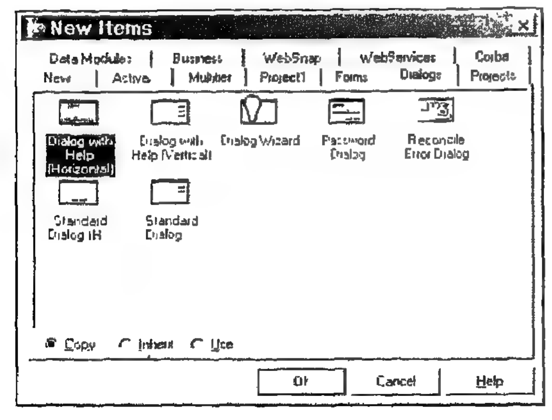
QuickReport Labels	مـن خلال هذا العنصر يتم بطريقة سريعة إنشاء الحقول التى يتألف منها أى تقرير كما هو موضح بالشكل التالى : QRLabelForm MasterTable Put data components on this band
QuickReport List	مـن خـلال هـذا العنصر يمكن إنشاء تقرير ملخص بشكل سريع من خلال النافذة الموضحة فى الشكل التالى : QRListForm (Report title) Put column headings on this band Put data components on this band Page (Page#)

QuickReport Master/Detail	مـن خـلال هـذا العنصر يمكن إنشاء تقارير تفصيلية وأساسية بشكل سريع وذلك من خلال النافذة الموضحة فى الشكل التالى : QRMDForm (Report title)
Tabbed Pages	مـن خـلال هـذا العنصر يتم إنشاء فورمة تتألف من عدة تبويبات أو صفحات كالفورمة الموضحة فى الشكل التالى : Tabbed Notebook Dialog TabSheet1 TabSheet2 TabSheet3 OK Cancel Help

الصفحة Dialogs بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



من خلال العناصر الموجودة فى هذه الصفحة يمكن أنشاء أنواع كثيرة من صناديق الحوار.

الجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
Dialog With Help (Horizontal)	من خلال هذا العنصر يتم إنشاء صندوق حوار يكون مشتمل على مجموعة من المفاتيح بجوار بعضها البعض فى الاتجاه الأفقى كما هو موضح بالشكل التالى :
Dialog With Help (Vertical)	من خلال هذا العنصر يتم إنشاء صندوق حوار يكون مشتمل على مجموعة من المفاتيح فوق بعضها البعض فى الاتجاه الرأسى كما هو موضح بالشكل التالى :

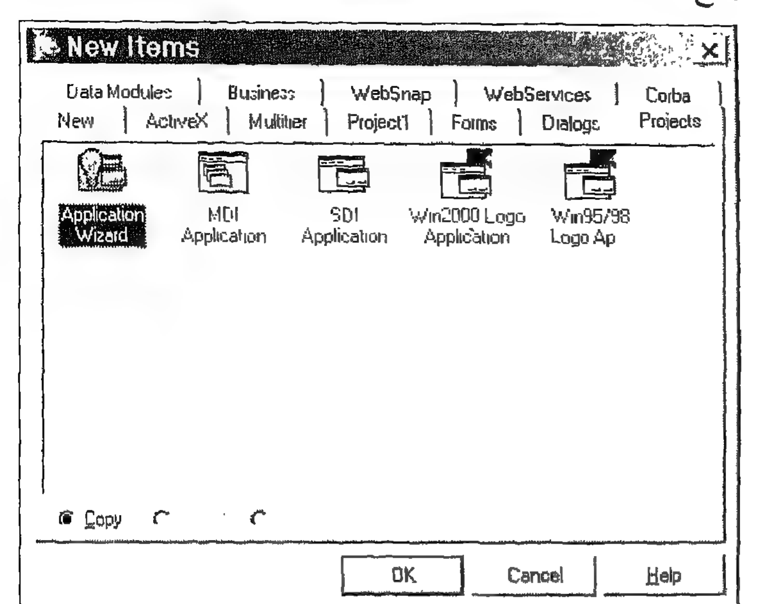
	: Dialog OK Cancel Help
Dialog Wizard	مـن خـلال هـذا العنصـر يـتم إنشـاء معـالج Wizard يضـم عدد من صـناديق الحـوار المتـتابعة والـتى يـتم التنقل بينها من خلال المفتاح Next كما هو موضح بالشكل التالى : Dialog Wizard Select dialog style: Single page dialog Multipage, using PageControl Next > Cancel
Password Dialog	مـن خـلال هـذا العنصـر يـتم إنشـاء صـندوق حـوار يكـون مخصص لاستقبال كلمة السر من المستخدم كما هو موضح بالشكل التالى : : Password Dialog Enter password: OK Cancel
Reconcile Error Dialog	مـن خـلال هذا العنصر يتم إنشاء صندوق حوار يتم من خلاله عرض مجموعـة من رسائل الخطأ التى يتم تحديثها كما هو موضح بالشكل التالى :

العنصر	الوصف
Standard Dialog (Horizontal)	من خلال هذا العنصر يتم إنشاء صندوق حوار قياسى لا يشتمل على المفتاح Help كما المفاتيح التى به تكون مرصوصة أفقيا كما هو موضح بالشكل التالى :
Standard Dialog (Vertical)	من خلال هذا العنصر يتم إنشاء صندوق حوار قياسى لا يشتمل على المفتاح Help كما المفاتيح التى به تكون مرصوصة رأسيا كما هو موضح بالشكل التالى :

الصفحة Projects بصندوق الحوار New Items

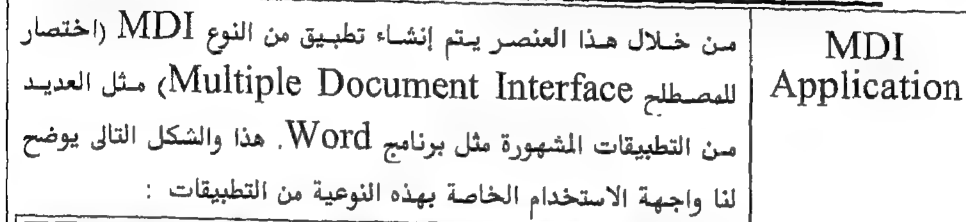
الشكل التالى يوضح لنا محتويات هذه الصفحة :

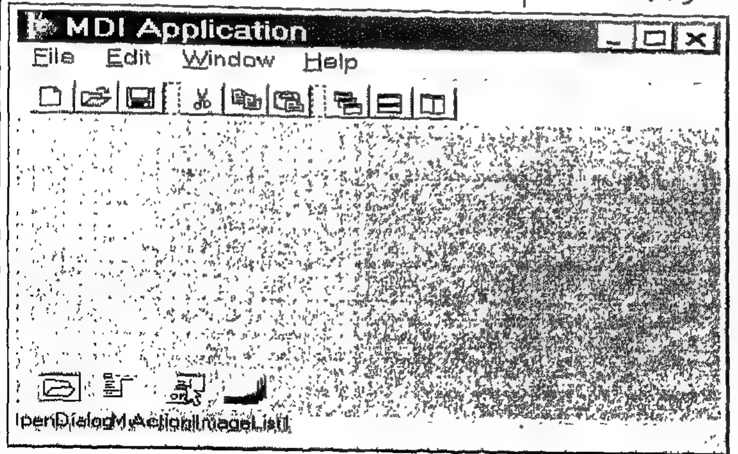
شكل توضيحى :



من خـلال العناصر الموجـودة بهذه الصفحة يمكن إنشاء أنواع عديدة من المشاريع والتطبيقات. هـذا والجـدول الـتالى يقـدم لـنا وصف مختصـر للعناصـر الموجـودة فـى هـذه الصفحـــة :

العنصر	الوصف والاستخدام
Application Wizard	مـن خلال هذا العنصر يتم إنشاء تطبيق بطريقة سهلة وبسيطة وذلك عن طريق التعامل مع معالج التطبيق Application Wizard الموضح فى الشكل التالى : Application Wizard Your application can contain a menu bar by checking one or more of the following standard Windows menus: File menu The file menu contains items such as Open, Save and Exit. Edit menu The Edit menu contains Undo, Cut, Copy and Paste items Window menu The Window menu contains window management functions specifically designed for MDI applications Help menu The Help menu provides access to your application's help file as well as an item to show the About box Next > Cancel
CLX MDI Application	مـن خـلال هـذا العنصـر يـتم إنشـاء تطبـيق من النوع MDI (اختصار للمصـطلح Multiple Document Interface) مـثل العديد مـن التطبـيقات المشـهورة مـثل برنامج Word. والتطبيق الذى نحصل علـيه يمكـن تشـغيله من خلال بيئة الويندوز والعديد من أنظمة التشغيل الأخـرى مثل Linux. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات : MDI Application File Edit Window Help



<table>
<tr>
<td>MDI Application</td>
<td>من خلال هذا العنصر يتم إنشاء تطبيق من النوع MDI (اختصار للمصطلح Multiple Document Interface) مثل العديد من التطبيقات المشهورة مثل برنامج Word. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات :

</td>
</tr>
<tr>
<td>SDI Application</td>
<td>من خلال هذا العنصر يتم إنشاء تطبيق من النوع SDI (اختصار للمصطلح Single Document Interface) مثل برنامج WordPad الذى يأتى مع بيئة الويندوز. هذا والشكل التالى يوضح لنا واجهة الاستخدام الخاصة بهذه النوعية من التطبيقات :
SDI Application
File Edit Help</td>
</tr>
</table>

Win 2000 Logo Application	مـن خـلال هـذا العنصر يـتم إنشـاء تطبـيقات تحمل بعض صفات بيئة ويندوز ٢٠٠٠ ومثل هذه التطبيقات يكون لها واجهة استخدام كالموضحة فى الشكل التالى : Windows2000 Client Application File Edit Help MainMActionImageOpenDSaveDPopupMenu1
Win 95/98 Logo Application	مـن خـلال هـذا العنصر يـتم إنشـاء تطبـيقات تحمل بعض صفات بيئة ويـندوز ٩٥ أو ٩٨ ومـثل هـذه التطبـيقات يكـون لهـا واجهـة اسـتخدام كالموضحة فى الشكل التالى : Win95/98 Logo Application File Edit Help MainMActionImageOpenDSaveDialog

الصفحة Data Modules بصندوق الحوار New Items

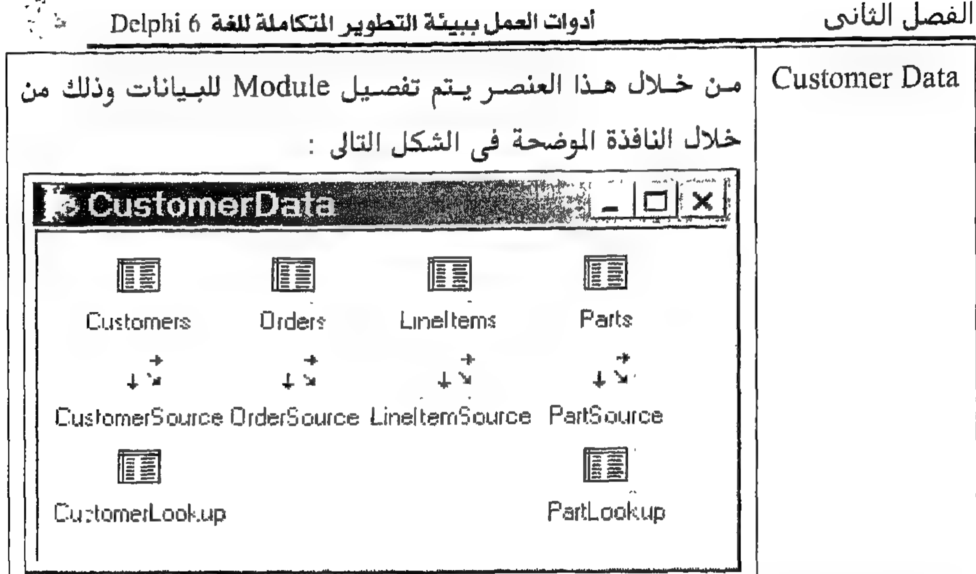
الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :

من خلال العناصر الموجودة فى هذه الصفحة نستطيع إنشاء نوعين مختلفين من Modules البيانات. هذا والجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
CLX Employee Data	من خلال هذا العنصر يتم إنشاء Module لتوظيف البيانات فى التطبيقات التى من النوع CLX وهى التطبيقات التى تكون مهيئة للعمل بكل من بيئة الويندوز وأنظمة التشغيل الأخرى مثل Linux. هذا ويتم التعامل مع هذا العنصر من خلال النافذة الموضحة فى الشكل التالى :



Customer Data	مـن خـلال هـذا العنصـر يـتم تفصـيل Module للبـيانات وذلك من خلال النافذة الموضحة فى الشكل التالى : 

الصفحة Business بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :

مـن خـلال العناصـر الموجـودة فـى هـذه الصـفحة يمكـن إنشـاء أنـواع مختلفة من التطبـيقات الـتجارية مـثل تطبيقات قواعد البيانات وتطبيقات الويب وتطبيقات دعم إتخاذ القـرار وتطبـيقات الإعـداد السريع لتقارير العمل بالإضافة إلى تطبيقات التمثيل البيانى. هذا والجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

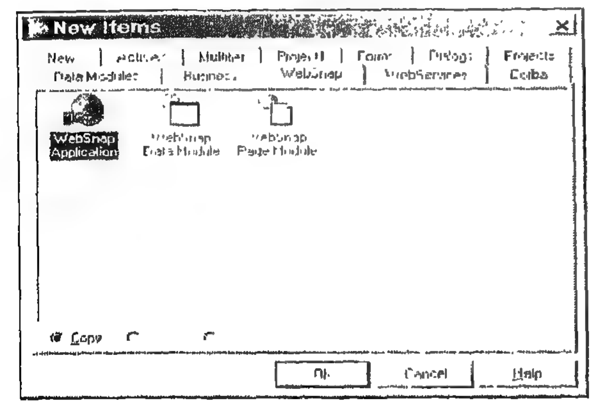
العنصر	الوصف والاستخدام
Database Form Wizard	من خلال العنصر يتم إنشاء فورمة لتطبيق قواعد بيانات وذلك من خلال المعالج Database Form Wizard الذى يظهر على الشاشة والموضح بالشكل التالى : Database Form Wizard What type of form do you want the wizard to create? Form Options Create a simple form Create a master/detail form DataSet Options Create a form using TTable objects Create a form using TQuery objects Help Next > Cancel
DB Web Application Wizard	من خلال العنصر يتم إنشاء تطبيق قواعد بيانات يعمل بشبكة الويب وذلك من خلال المعالج DB Web Application Wizard الموضح فى الشكل التالى : Database Web Application Wizard What type of web application do you want to create? Web Application Type ISAPI/NSAPI Dynamic Link Library CGI Stand-alone Executable Win-CGI Stand-alone Executable DataSet Options Create a web module using a TTable object Create a web module using a TQuery object Next > Cancel
Decision Cube Sample	من خلال العنصر يتم إنشاء نموذج مبسط للتطبيقات التى تعمل على دعم إتخاذ القرار. هذا والشكل التالى يوضح لنا النموذج الذى نحصل عليه من هذا العنصر :

العنصر	الوصف
QuickReport Wizard	من خلال العنصر يتم إنشاء تقارير عمل سريعة وذلك من خلال المعالج QuickReport Wizard الموضح فى الشكل التالى :
TeeChart Wizard	من خلال العنصر يتم إنشاء أشكال ورسومات بيانية وذلك من خلال المعالج TeeChart Wizard الموضح فى الشكل التالى :

الصفحة WebSnap بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



من خلال العناصر الموجودة فى هذه الصفحة يمكن إنشاء أنواع مختلفة من تطبيقات التى تعمل فى شبكة الويب. هذا والجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
WebSnap Application	عن طريق هذا العنصر يمكن إنشاء تطبيقات تعمل فى شبكة الويب وذلك من خلال صندوق الحوار New WebSnap Application الموضح فى الشكل التالى :

WebSnap Data Module	عن طريق هذا العنصر يمكن إنشاء Module للبيانات يتم استخدامه مع التطبيقات التى تعمل بشبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New WebSnap Data Module الموضح فى الشكل التالى : New WebSnap Data Module Module Options Creation: On Demand Caching: Cache Instance OK Cancel Help
WebSnap Page Module	عن طريق هذا العنصر يمكن إنشاء WebSnap Page Module لاستخدامها مع التطبيقات التى تعمل بشبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New WebSnap Page Module الموضح فى الشكل التالى : New WebSnap Page Module Producer Type: PageProducer Script Engine: JScript HTML New File: ☑ Template: Standard Page Name: PageProducerPage2 Title: PageProducerPage2 Published ☑ Login Required: ☐ Module Options Creation: On Demand Caching: Cache Instance ☐ Default OK Cancel Help

الصفحة Web Services بصندوق الحوار New Items

الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :

New Items
New | ActiveX | Multitier | Project1 | Forms | Dialogs | Projects
Data Modules | Business | WebSnap | WebServices | Corba
Soap Server Application | Soap Server Data Module | Web Service Importer
Copy
OK | Cancel | Help

من خلال العناصر الموجودة فى هذه الصفحة يمكن إنشاء تطبيقات وخدمات تعمل بخوادم شبكة الويب . هذا والجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

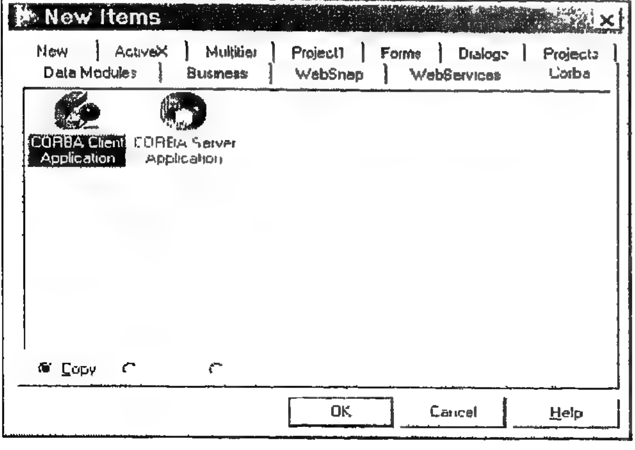
العنصر	الوصف والاستخدام
Soap Server Application	عن طريق هذا العنصر يمكن إنشاء تطبيق يعمل بأى من خوادم شبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار New Soap Server Application الموضح فى الشكل التالى : New Soap Server Application You may select from one of the following types of World Wide Web server applications. ISAPI/NSAPI Dynamic Link Library CGI Stand-alone executable Win-CGI Stand-alone executable Apache Shared Module (DLL) Web App Debugger executable OK | Cancel | Help

Soap Server Data Module	عن طريق هذا العنصر يمكن إنشاء Module بيانات للتطبيقات التى تعمل بخوادم شبكة الويب. وهذا الإنشاء يتم من خلال المعالج Soap Data Module Wizard الموضح فى الشكل التالى : Soap Data Module Wizard Class Name: OK Cancel Help
Web Services Importer	عن طريق هذا العنصر يمكن إنشاء أداة لاستيراد خدمات الويب وإضافتها للتطبيق الذى يعمل بخوادم شبكة الويب. وهذا الإنشاء يتم من خلال صندوق الحوار Web Services Import الموضح فى الشكل التالى : Web Services Import Import Advanced WSDL or XML Schema Location (Filename or URL) Browse... Cancel Help

الصفحة CORBA بصندوق الحوار New Items

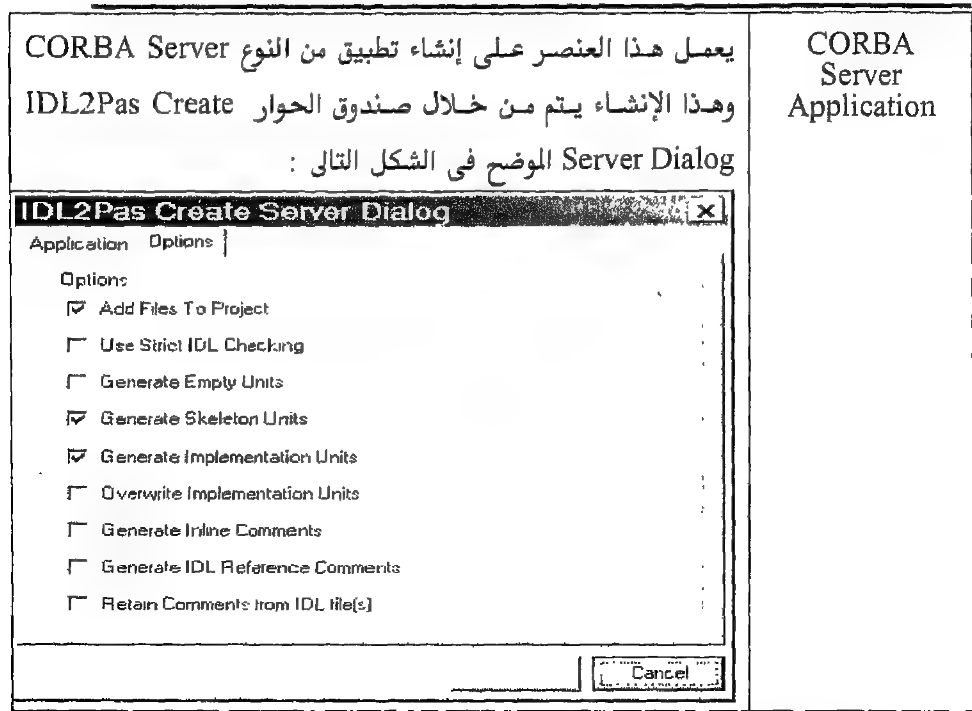
الشكل التالى يوضح لنا محتويات هذه الصفحة :

شكل توضيحى :



من خلال العناصر الموجودة فى هذه الصفحة يمكن إنشاء تطبيقات CORBA سواء كانت للعملاء Client أو للخادم Server. هذا والجدول التالى يقدم لنا وصف مختصر للعناصر الموجودة فى هذه الصفحة :

العنصر	الوصف والاستخدام
CORBA Client Application	يعمل هذا العنصر على إنشاء تطبيق من النوع CORBA Client وهذا الإنشاء يتم من خلال صندوق الحوار IDL2Pas Create Client Dialog الموضح فى الشكل التالى :

CORBA Server Application	يعمل هذا العنصر على إنشاء تطبيق من النوع CORBA Server وهذا الإنشاء يتم من خلال صندوق الحوار IDL2Pas Create Server Dialog الموضح فى الشكل التالى : 

الأمر Open بالقائمة File

يعمل هذا الأمر على عرض صندوق الحوار Open (الموضح فى الشكل التالى) من أجل فتح وتحميل مشروع أو فورمة أو وحدة تم إنشاؤها قبل ذلك كما يمكن أيضا فتح ملف نصى داخل محرر الكود البرمجى Code Editor :

شكل توضيحى :

الأمر Open Project بالقائمة File

يعمل هذا الأمر على عرض صندوق الحوار Open Project (الموضح في الشكل التالى) والذى يمكن من خلاله فتح وتحميل أى مشروع من المشروعات التى سبق إنشاؤها (سواء كانت مخزنة فى ملف بالامتداد BPK. أو الامتداد BPR.) :

شكل توضيحى :



يمكن أيضا فتح صندوق الحوار Open Project عن طريق الضغط على المفتاحين Ctrl+F11 بلوحة المفاتيح.

الأمر Reopen بالقائمة File

من خلال هذا الأمر يمكن إعادة فتح أى مشروع أو Module من المشاريع أو الـ Modules التى سبق فتحها قريبا والتى يتم عرض أسماؤها ومواقعها فى القائمة الفرعية الخاصة به كما هو موضح بالشكل التالى :

شكل توضيحى :



الأمر Save بالقائمة File

عـن طريق هذا الأمر يتم حفظ الملف الذى يتم التعامل معه حاليا وهذا الحفظ يكون بنفس الأسم المخصص حاليا لهذا الملف

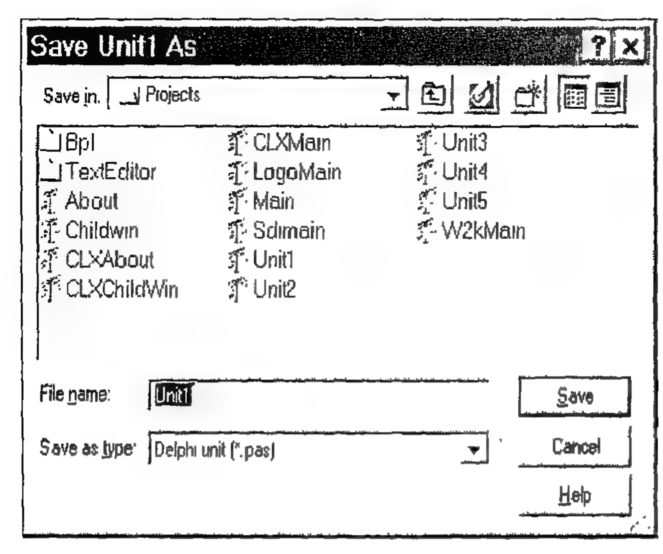


يمكن استدعاء الأمر Save عن طريق الضغط على المفتاحين Ctrl+S بلوحة المفاتيح.

الأمر Save As بالقائمة File

مـن خلال هذا الأمر يتم حفظ الملف الحالى باستخدام اسم جديد مما يؤدى إلى عمل نسـخة مـن هذا الملف وهذه النسخة الجديدة تكون متضمنة كافة التعديلات التى تم إجرؤها مؤخـرا على ملفات المشروع. وهذا الحفظ يتم من خلال صندوق الحوار Save As الموضح فى الشكل التالى :

شكل توضيحى :



الأمر Save Project As بالقائمة File

يعمـل هـذا الأمـر عـلى حفـظ المشـروع الحـالى بأسم جديد وذلك من خلال صندوق الحوار Save Project As الموضح بالشكل التالى :

شكل توضيحى :

الأمر Save All بالقائمة File

من خـلال هـذا الأمر يمكن حفظ كافة الملفات المفتوحة سواء كانت تخص المشروع الحالة أو الـModules.



يمكن استدعاء الأمر Save All عن طريق الضغط على مجموعة المفاتيح Shift+ Ctrl+S بلوحة المفاتيح.

الأمر Close بالقائمة File

يعمل هذا الأمر على إغلاق المشروع الحالى وكافة الوحدات والفورم المرتبطة به.

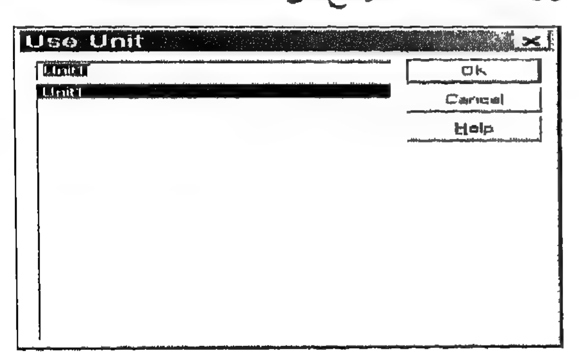
الأمر Close All بالقائمة File

يعمل هذا الأمر على إغلاق كافة الملفات المفتوحة.

الأمر Use Unit بالقائمة File

يعمل هذا الأمر على إضافة الوحدة المختارة إلى الجملة uses الموجودة بالـModule النشط حاليا وذلك من خلال صندوق الحوار Use Unit الموضح فى الشكل التالى :

شكل توضيحى :







يمكن أيضا فتح صندوق الحوار Use Unit عن طريق الضغط على المفتاحين Alt+F11 بلوحة المفاتيح.

الأمر Print بالقائمة File

يقوم هذا الأمر بإرسال ملف الفورمة النشطة حاليا إلى الطابعة لطباعته وهذا الإرسال يتم من خلال صندوق الحوار Print Form الموضح فى الشكل التالى :

شكل توضيحى :

Print Form

Form printing options

Proportional (based on PixelsPerInch)

Print to fit page

No scaling

Setup... OK Cancel Help

الأمر Exit بالقائمة File

يعمل هذا الأمر على غلق المشروع المفتوح وإنهاء التعامل مع لغة Delphi.

القائمة Edit

يتم استخدام مجموعة الأوامر الموجودة فى القائمة Edit للتعامل مع النصوص والمكونات فى مرحلة التصميم.

الجدول التالى يقدم لنا وصفا مختصرا للأوامر الموجودة بهذه القائمة :

الأمر	الوصف والاستخدام
Undo/Undelete	هذا الأمر يلغى أثر آخر فعل قمت به كما يعيد العنصر الذى تم مسحه أخيرا.
Redo	هذا الأمر يلغى تأثير الأمر Undo أو الأمر Undelete.
Cut	من خلال الأمر يتم استقطاع العنصر المختار من موضعه ويضعه فى لوحة الاقتباس clipboard.

Copy	يعمل هذا الأمر على وضع نسخة من العنصر المختار بلوحة الاقتباس وفى نفس الوقت يترك الأصل فى موضعه.
Paste	يعمل هذا الأمر على أخذ نسخة من محتويات لوحة الاقتباس ووضعها فى محرر نافذة الكود البرمجى Code Editor أو فى داخل الفورمة.
Delete	من خلال هذا الأمر يتم مسح العنصر المختار.
Select All	يؤدى هذا الأمر إلى اختيار كافة المكونات الموجودة فى الفورمة.
Align to Grid	يعمل هذا الأمر على ضبط المكونات المختارة على أقرب نقطة فى شبكة النقط بالفورمة.
Bring to Front	من خلال هذا الأمر يتم نقل المكون المختار ليصبح فوق المكونات الأخرى وذلك فى حالة وجود مكونات بعضها فوق بعض.
Send to Back	من خلال هذا الأمر يتم نقل المكون المختار ليصبح تحت المكونات الأخرى وذلك فى حالة وجود مكونات بعضها فوق بعض.
Align	يعمل هذا الأمر على ضبط المكونات الموجودة فى الفورمة.
Size	عن طريق هذا الأمر يمكن تغيير حجم المكونات الموجودة فى الفورمة.
Scale	يعمل هذا الأمر على تغيير حجم كافة المكونات الموجودة فى الفورمة.
Tab Order	من خلال هذا الأمر يتم التعديل فى ترتيب انتقال دفة التحكم focus بين المكونات (باستخدام المفتاح Tab الموجود بلوحة المفاتيح) الموجودة فى القورمة النشطة.

Creation Order	يعمل هذا الأمر على تعديل الترتيب الذى تم من خلاله إنشاء المكونات الغير مرئية (فى مرحلة التشغيل Run-Time).
Flip Children	عن طريق هذا الأمر يتم عكس أو قلب تخطيط أدوات التحكم من اليمين إلى اليسار أو العكس كما لو كنا ننظر إليها فى مرآة.
Lock Controls	يعمل هذا الأمر على تأمين كافة المكونات الموجودة بالفورمة بحيث لا يمكن نقلهم من مواضعهم الحالية.
Add to Interface	من خلال هذا الأمر يمكن تعريف أسلوب أو حدث أو خاصية جديدة لأى أداة من أدوات التحكم ActiveX.

الأمر Undo/Delete بالقائمة Edit

فى أثناء العمل بمحرر الكود البرمجى Code Editor يمكن أن نختار الأمر Undo من القائمة Edit وذلك لإلغاء تأثير أحدث استخدام للوحة المفاتيح أو لأحدث فعل تم بالماوس. أما عند التعامل مع فورمة فإنه يتم اختيار الأمر Delete من القائمة Edit لاسترجاع آخر عنصر تم مسحة من الفورمة.

استخدام الأمر Undo فى محرر الكود البرمجى Code Editor

الأمر Undo يعمل على إعادة إدراج أى حروف قمت بمسحها كما يسمح أى حروف قمت بكتابتها كما يقوم أيضا باستبدال أى حروف قمت بتخطيها overwrite أو قد يعمل على جعل مؤشر الكتابة يعود مرة أخرى إلى موضعه السابق.

تستطيع ان تتراجع عن العديد من الأفعال التى قمت بها تباعا وذلك عن طريق اختيار الأمر Undo بصفة متكررة مما يؤدى إلى إزالة التغييرات التى أجريتها من خلال مبدأ "الرجوع للخلف تباعا" عبر الأفعال التى قمت بها والعودة مرة أخرى إلى الحالة السابقة.

لكى تقرر الحد الذى يقف عنده الأمر Undo :

(١) افـتح القائمـة Tools واخـتر منها الأمر Editor Options ليظهر على الشاشة صندوق الحوار Editor Options الموضح فى الشكل التالى :

شكل توضيحى :

Editor Properties

General | Display | Key Mappings | Color | Code Insight

Editor options

- [x] Auto indent mode
- [x] Insert mode
- [ ] Use tab character
- [ ] Smart tab
- [ ] Optimal fill
- [x] Backspace unindents
- [x] Cursor through tabs
- [x] Group undo
- [ ] Cursor beyond EOF
- [ ] Undo after save
- [ ] Keep trailing blanks
- [ ] BRIEF regular expressions
- [ ] Persistent blocks
- [x] Overwrite blocks
- [ ] Double click line
- [x] Find text at cursor
- [ ] Force cut and copy enabled
- [x] Use syntax highlight

Editor SpeedSetting: Default keymapping

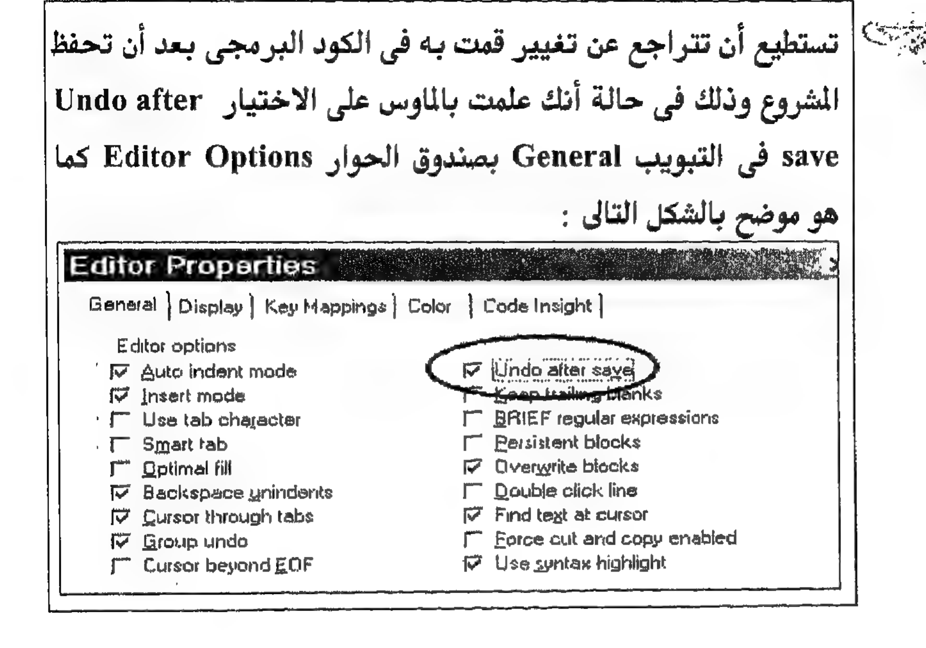
Block indent: 2 Undo limit: 32767

Tab stops: 8

Syntax extensions: pas;dpr;dpk;inc;dfm;xfm

OK Cancel Help

(٢) انقـر بالمـاوس عـلى الحقـل الخاص بالقائمة المنسدلة Undo limit وفيه اكتب القيمة العدديـة الـتى تمـثل الحـد الأقصـى الـذى ترغـبه للأفعـال التى يمكن التراجع عنها من خلال الأمر undo.



تستطيع أن تتراجع عن تغيير قمت به فى الكود البرمجى بعد أن تحفظ المشروع وذلك فى حالة أنك علمت بالماوس على الاختيار Undo after save فى التبويب General بصندوق الحوار Editor Options كما هو موضح بالشكل التالى :



لـو أنـك تراجعـت عـن العمليـة التى تعرف بـ block operation فى هذه الحالة يظهر الملف الذى تتعامل معه كما كان فى السابق قبل أن تقوم بتنفيذ هذه العملية.



الأمر undo لا يعمل على تغيير اختيار تم تحديده ويكون له تأثير على أكثر من نافذة.

لكى تتراجع عن مجموعة من الأفعال قم بالآتى :

(١) افـتح القائمـة Tools واخـتر مـنها الأمر Editor options ليظهر على الشاشة صندوق الحوار Editor Options.

(٢) علم بالماوس على الاختيار Group Undo كما هو موضح بالشكل التالى :

شكل توضيحى :

Editor Properties

General | Display | Key Mappings | Color | Code Insight

Editor options

- [x] Auto indent mode
- [x] Insert mode
- [ ] Use tab character
- [ ] Smart tab
- [ ] Optimal fill
- [x] Backspace unindents
- [x] Cursor through tabs
- [x] Group undo
- [ ] Cursor beyond EOF
- [x] Undo after save
- [ ] Keep trailing blanks
- [ ] BRIEF regular expressions
- [ ] Persistent blocks
- [x] Overwrite blocks
- [ ] Double click line
- [x] Find text at cursor
- [ ] Force cut and copy enabled
- [x] Use syntax highlight

Editor SpeedSetting: Default keymapping

Block indent: 2 Undo limit: 32767

Tab stops: 8

Syntax extensions: pas;dpr;dpk;inc;dfm;xfm

OK Cancel Help

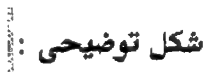
الأمر Redo بالقائمة Edit

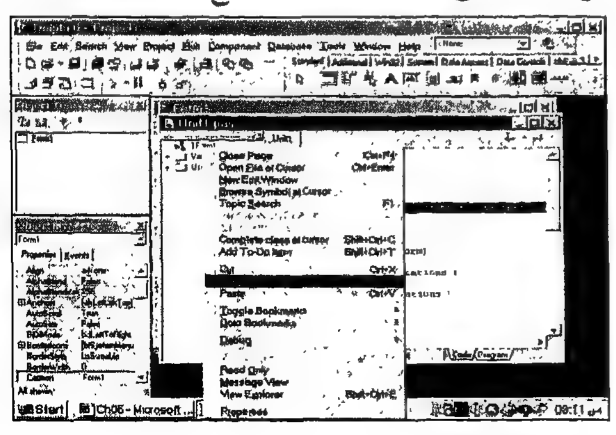
يـتم استخدام الأمر Redo من القائمة Edit لعكس التأثيرات التى نجمت عن آخر اسـتخدام للأمر Undo. هذا والأمر Redo يكون له تأثير عندما يتم استخدامه بعد استخدام الأمـر Undo مباشـرة. هـذا ويكـون الأمـر Redo غـير مـتاح للاسـتخدام لعكـس الـتأثيرات الناجمة عن الأمر Undelete.

الأمر Cut بالقائمة Edit (أو بالقائمة المختصرة بنافذ محرر الكود البرمجى)

اختيار الأمر Cut من القائمة Edit يؤدى إلى إزالة العناصر التالية من الموضع الحالة لها ووضعهم فى لوحة الاقتباس clipboard :

- النص المختار أى المعلم عليه فى محرر الكود البرمجى Code Editor (وأنت تستطيع أيضا النقر بالزر الأيمن للماوس على النص المعلم عليه ثم اختيار الأمر Cut من القائمة المختصرة التى تظهر على الشاشة كما هو موضح بالشكل التالى :


شكل توضيحى :



- المكونات المختارة فى الفورمة النشطة.
- القوائم المختارة فى صندوق الحوار Menu Designer.

هذا ويعمل الأمر Cut على استبدال المحتويات الحالية بلوحة الاقتباس بالعنصر المختار. وعلى العموم لكى تضع محتويات لوحة الاقتباس فى أى موضع ترغبه بالمشروع فى هذه الحالة افتح القائمة Edit ثم اختر منها الأمر Paste.

الأمر Copy بالقائمة Edit (أو بالقائمة المختصرة بنافذ محرر الكود البرمجى)

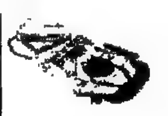
يمكن اختيار الأمر Copy من القائمة Edit (أو النقر بالزر الأيمن للماوس فى محرر الكود البرمجى Code Editor ثم اختيار الأمر Copy من القائمة المختصرة التى تظهر على الشاشة) وذلك لوضع نسخة طبق الأصل من النص المختار أو مكون مختار أو قائمة مختارة فى لوحة الإقتباس وترك الأصل بدون أن يحدث له أى تغيير. هذا والأمر Copy يستبدل المحتويات الحالية بلوحة الإقتباس بالعناصر المختارة.

هذا ولكى تضع محتويات لوحة الإقتباس بأى موضع عليك إذن أن تستخدم الأمر Paste من القائمة Edit.



الأمر Paste بالقائمة Edit (أو بالقائمة المختصرة بنافذ محرر الكود البرمجي)

يمكن اختيار الأمر Paste من القائمة Edit وذلك لإدراج محتويات لوحة الاقتباس داخل صفحة محرر الكود البرمجي Code Editor النشط أو الفورمة النشطة أو القائمة النشطة في صندوق الحوار Menu Designer.



تستطيع أن تلصق paste النصوص فقط فى نافذة محرر الكود البرمجى Code Editor أما المكونات فيتم لصقها فى الفورمة فقط فى حين أن عناصر القوائم يتم لصقها فى صندوق الحوار Menu Designer فقط.

عند إجراء عملية اللصق داخل نافذة محرر الكود البرمجى Code Editor فإنه يتم إدراج النص عند الموضع الموجود به مؤشر الكتابة. وأنت تستطيع النقر بالزر الأيمن للماوس فى محرر الكود البرمجي Code Editor ثم اختيار الأمر Paste من القائمة المختصرة التى تظهر على الشاشة.

عند إجراء عملية اللصق داخل فورمة فى هذه الحالة نجد أن المكونات الغير مرئية (فى مرحلة التشغيل Run-Time) يتم لصقها بالركن الأيسر العلوى للفورمة أما المكونات الظاهرة (فى مرحلة التشغيل Run-Time) فيتم لصقها فى نفس الموضع الذى تم فيه نسخ أو قص هذه المكونات.

عند إجراء عملية اللصق داخل صندوق الحوار Menu Designer فإنه يتم إدراج عناصر القوائم عند الموضع الموجود به مؤشر الماوس.

هذا وأنت تستطيع لصق المحتويات الحالية للوحة الاقتباس أى عدد من المرات حتى تقوم بعمل قص أو نسخ عنصر جديد داخل لوحة الاقتباس.

الأمر Delete بالقائمة Edit

يتم استخدام الأمر Delete من القائمة Edit لمسح النص المختار أو المكون المختار بدون وضع نسخه منه فى بوحة الاقتباس ومن ثم فإنك لا تستطيع لصق العنصر الممسوح ولكن بالرغم من ذلك فإن تستطيع استعاده هذا العنصر مرة أخرى بأن تستخدم الأمر Undelete من القائمة Edit مباشرة بعد استخدام الأمر Delete من القائمة Edit.

المسح يكون مفيد لو أنك ترغب فى إزالة عنصر وفى نفس الوقت لا ترغب فى استبدال محتويات لوحة الاقتباس بهذا العنصر.

الأمر Select All بالقائمة Edit

يعمل الأمر Select All بالقائمة Edit على اختيار كل عنصر موجود فى النافذة النشطة. هذا وفى أداة تصميم الفورمة يتم استخدام الأمر Select All من القائمة Edit لاختيار كل مكون موجود فى الفورمة. وعندما تختار أكثر من مكون فى نفس الوقت تجد أن الخصائص المشتركة بين هذه المكونات هى التى تظهر فقط فى النافذة Object Inspector.

أما فى نافذة محرر الكود البرمجى Code Editor فإنه يتم استخدام الأمر Select All من القائمة Edit لاختيار كل النص الموجود فى الملف المعروض حاليا فى نافذة المحرر.

الأمر Align to Grid بالقائمة Edit (أو من القائمة المختصرة بالفورمة)

يتم اختيار الأمر Align to Grid من القائمة Edit لضبط المكونات المختارة على أقرب نقطة فى شبكة النقط الموجودة فى الفورمة بمود التصميم.

هذا ولكن تختار أكثر من مكون فى نفس الوقت فإما أن تضغط على المفتاح Shift بلوحة المفاتيح فى أثناء النقر بالماوس على كل مكون على حدة أو ترسم بالماوس مستطيل اختيار حول مجموعة المكونات التى ترغب فى اختيارها.

افتح القائمة Tools ثم اختر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :

شكل توضيحى :

Environment Options
Type Library | Environment Variables | Delphi Direct | Internet
Preferences | Designer | Object Inspector | Palette | Library | Explorer
Autosave options
Editor files
Project desktop
Desktop contents
Desktop only
Desktop and symbols
Docking
Auto drag docking
Compiling and running
Show compiler progress
Warn on package rebuild
Minimize on run
Hide designers on run
Shared repository
Directory:
Browse...
OK
Cancel
Help



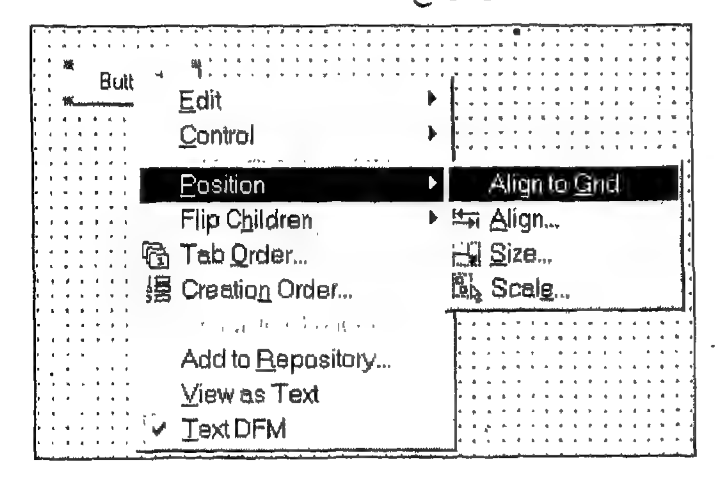
وفى التبويب Designer علم على الاختيار Snap to Grid كما هو موضح بالشكل التالى

شكل توضيحى :

ومـن ثـم فإن أى مكون سيضبط نفسه تلقائيا على نقط الشبكة عند إضافته للفورمة.
كذلـك تسـتطيع تحديـد حجـم شبكة النقط التى بالفوركة وذلك عن طريق كل من العداد X والعداد Y فى التبويب Designer بصندوق الحوار Environment Options.

تسـتطيع أن تصـل إلى الأمـر Align to Grid عـن طـريق الأمـر Position الموجود بالقائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة النشطة أو Module بيانات مع المكونات المختارة كما هو موضح بالشكل التالى :

شكل توضيحى :-



الأمر Bring to Front بالقائمة Edit (أو بالقائمة المختصرة بالفورمة)

من خلال الأمر Bring to Front الموجود بالقائمة Edit يتم نقل المكون المختار ليصبح فوق كافة المكونات الأخرى الموجودة فى الفورمة. ومثل هذا العمل يطلق عليه التغيير فى الترتيب العمودى للمكونات فى اتجاه محور Z.



لا يعمل الأمر Bring to Front فى حالة أنك اخترت خليط من المكونات (أدوات التحكم) بعضها له نافذة Windowed والآخر ليس له نافذة non-windowed. فعلى سبيل المثال لا تستطيع تغيير الترتيب العمودى Z-Order للكائن label فى أثناء اختياره مع الكائن button فى نفس الوقت.

الأمر Send to Back بالقائمة Edit (أو بالقائمة المختصرة بالفورمة)

من خلال الأمر Send to Back الموجود بالقائمة Edit يتم نقل المكون المختار ليصبح تحت كافة المكونات الأخرى الموجودة فى الفورمة. ومثل هذا العمل يطلق عليه التغيير فى الترتيب العمودى للمكونات فى اتجاه محور Z.



لا يعمل الأمر Send to Back فى حالة أنك اخترت خليط من المكونات (أدوات التحكم) بعضها له نافذة Windowed والآخر ليس له نافذة non-windowed. فعلى سبيل المثال لا تستطيع تغيير الترتيب العمودى Z-Order للكائن label فى أثناء اختياره مع الكائن button فى نفس الوقت.

الأمر Align بالقائمة Edit

اختيار الأمر Align من القائمة Edit يؤدى إلى فتح صندوق الحوار Alignment الموضح فى الشكل التالى :

شكل توضيحى :

Alignment

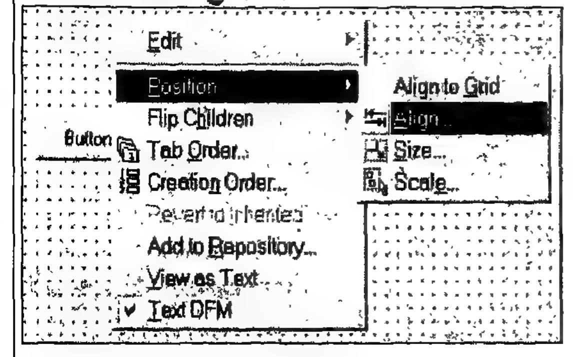
Horizontal: No change, Left sides, Centers, Right sides, Space equally, Center in window

Vertical: No change, Tops, Centers, Bottoms, Space equally, Center in window

OK   Cancel   Help



تستطيع أيضا الوصول للأمر **Align** (ولصندوق الحوار **Alignment**) عن طريق الأمر **Position** الموجود بالقائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس بفورمة نشطة كما هو موضح بالشكل التالى :



صندوق حوار الضبط Alignment

يتم استخدام صندوق الحوار هذا من أجل عمل line up للمكونات المختارة فى علاقة بين كل مكون والآخر أو فى علاقة بينهم وبين الفورمة. هذا والإختيارات الخاصة بكل من الضبط الأفقى والضبط الرأسى عبارة عن الآتى :

الاختيار	الوصف والاستخدام
No Change	هذا الاختيار لا يفعل أى تغيير على ضبط المكون.
Left sides	يعمل هذا الاختيار على ضبط الحواف اليسرى للمكونات المختارة على خط واحد (فى الاتجاه الأفقى فقط).

Centers	يعمـل هـذا الاختـيار عـلى ضـبط مراكـز المكونات المختارة على خط واحـد.
Right sides	يعمـل هـذا الاختـيار عـلى ضـبط الحواف اليمنى للمكونات المختارة على خط واحد (فى الاتجاه الأفقى فقط).
Tops	يعمل هذا الاختيار على ضبط الحواف العليا للمكونات المختارة على خط واحد (فى الاتجاه الرأسى فقط).
Bottoms	يعمـل هـذا الاختـيار عـلى ضبط الحواف السفلية للمكونات المختارة على خط واحد (فى الاتجاه الرأسى فقط).
Space Equally	يعمـل هـذا الاختـيار عـلى ضـبط المكونـات المختارة على خط واحد بحيث تبعد عن بعضها البعض بمسافات متساوية.
Center in window	يعمـل هذا الاختيار على ضبط المكونات المختارة بالنسبة لمركز نافذة الفورمة.

الأمر Size بالقائمة Edit

اختيار الأمر Size من القائمة Edit يؤدى إلى فتح صندوق الحوار Size الموضح فى الشكل التالى :

شكل توضيحى :

Size

Width:	Height:
◉ No change	◉ No change
○ Shrink to smallest	○ Shrink to smallest
○ Grow to largest	○ Grow to largest
○ Width:	○ Height:

OK Cancel Help



صندوق الحوار Size

يمكـن استخدام صندوق الحوار Size لتغيير حجم العديد من المكونات المختارة معا ليكونوا كلهم بنفس الارتفاع أو العرض بالضبط.

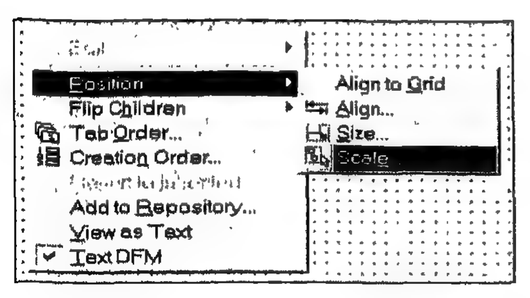
خـيارات تحديـد العـرض Width تعمـل عـلى تغيير الحجم الأفقى للمكونات المخـتارة. أمـا خـيارات تحديـد الإرتفـاع Height فـتعمل عـلى تغـيير الحجم الرأسى لـلمكونات المخـتارة. هـذا ويقـدم لـنا الجدول التالى الخيارات الخاصة بتحديد الحجم أفقيا أو رأسيا :

الاختيار	الوصف والاستخدام
No Change	هـذا الاختـيار لا يـؤدى إلى حـدوث أى تغـيير على حجم المكونات المختارة.
Shrink to smallest	يعمـل هـذا الاختـيار على تغيير حجم مجموعة من المكونات ليصبح ارتفاعهـا أو عرضـها مسـاويا لارتفـاع أو عـرض أصـغر مكـون فـى مجموعة المكونات.
Grow to largest	يعمـل هـذا الاختـيار على تغيير حجم مجموعة من المكونات ليصبح ارتفاعها أو عرضها مساويا لارتفاع أو عرض أكبر مكون فى مجموعة المكونات.
Width	مـن خـلال هـذا الاختـيار يمكـن تحديـد أى عرض نرغبه لمجموعة المكونات المختارة.
Height	مـن خـلال هـذا الاختـيار يمكـن تحديـد أى ارتفاع نرغبه لمجموعة المكونات المختارة.

الأمر Scale بالقائمة Edit (أو بالقائمة المختصرة الخاصة بالفورمة)

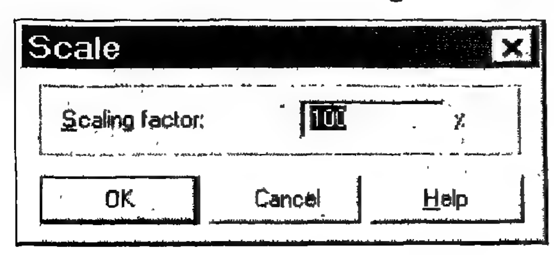
اختيار الأمر Scale من القائمة Edit أو من القائمة الفرعية الخاصة بالأمر Position الموجود بالقائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :



يؤدى ذلك إلى فتح صندوق الحوار Scale الموضح فى الشكل التالى :

شكل توضيحى :



صندوق الحوار Scale

يمكن استخدام صندوق الحوار Scale لتكبير أو تصغير حجم كافة المكونات الموجودة بالفورمة الحالية بنسبة معينة.

معامل التكبير والتصغير (نسبة مئوية)

فى الحقل المسمى Scaling Factor ادخل نسبة المئوية التى يتم من خلالها تكبير أو تصغير المكونات الموجودة بالفورمة علما بأنه إذا كانت القيمة أقل من ١٠٠٪ فإنها تؤدى إلى تصغير الحجم أما إذا كانت أكبر من ١٠٠٪ فإنها تعمل على تكبير الحجم.

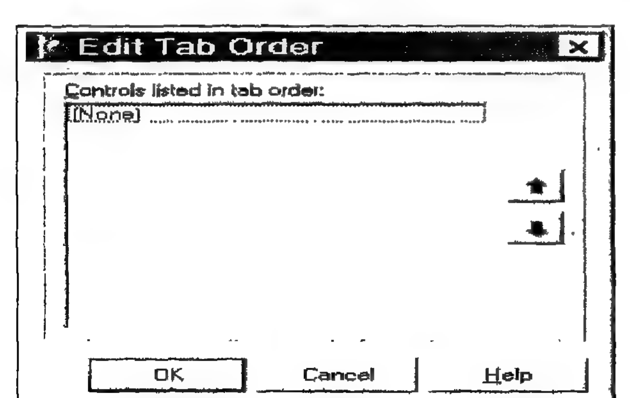


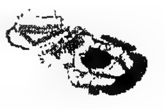
تتراوح قيمة المعامل Scaling Factor بين ٢٥٪ إلى ٤٠٠٪.

الأمر Tab Order بالقائمة Edit (أو بالقائمة المختصرة بالفورمة)

اختيار الأمر Tab Order من القائمة Edit يؤدى إلى فتح صندوق الحوار Edit Tab Order الموضح فى الشكل التالى :

شكل توضيحى :





تستطيع أيضا الوصول لصندوق الحوار Edit Tab Order عن طريق اختيار الأمر Tab Order من القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :

Form1
Edit
Position
Flip Children
Tab Order...
Creation Order...
Revert to Inherited
Add to Repository...
View as Text
Text DFM

صندوق الحوار Edit Tab Order

يمكـن اسـتخدام صـندوق الحوار Edit Tab Order للتعديل فى ترتيب التنقل بين المكونـات الموجـودة بالفورمـة (أو داخـل المكـون المخـتار فى حالة أنه يشتمل على مكونات أخرى داخله) باستخدام المفتاح Tab بلوحة المفاتيح فى مرحلة التشغيل Run-Time.

قائمة العرض Controls بصندوق الحوار Edit Tab Order

تشـتمل قائمـة العـرض Controls بصـندوق الحـوار Edit Tab Order مجموعـة المكونـات الموجودة فى الفورمة النشطة بحيث أن هذه المكونات تظهر فى هذه القائمة بنفس ترتيـب الوصـول إلـيها بالفورمـة. بمعـنى أن أول مكـون فـى هذه القائمة هو أول مكون يتم الـتعامل معـه فـى مـرحلة التشـغيل Run-Time وهكـذا...هـذا ويـتم تحديد ترتيب التنقل الافتراضى عن طريق الترتيب الذى تم به وضع المكونات فى الفورمة.

لكى تقوم بتغيير ترتيب التنقل الخاص بأى مكون اتبع الخطوات التالية :

(١) اخـتر اسـم المكـون الـذى ترغـبه فـى قائمـة العرض Controls كما هو موضح بالشكل التالى :

شكل توضيحى :

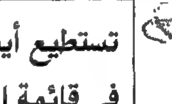
Edit Tab Order

Controls listed in tab order:

Edit1: TEdit
Button1: TButton
CheckBox1: TCheckBox

OK Cancel Help

(٢) انقر بالماوس على المفتاح Up [↑] لنقل المكون المختار لأعلى فى قائمة العرض Controls أو انقر بالماوس على المفتاح Down [↓] لنقله إلى أسفل فى قائمة العرض.



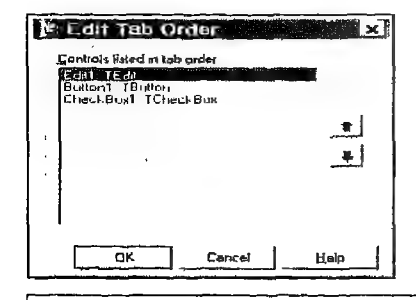
تستطيع أيضا سحب المكون المختار حتى تصل به إلى الموضع الذى ترغبه فى قائمة العرض Controls.

(٣) لحفظ التغييرات التى قمت به انقر بالماوس على المفتاح Ok.

الأمر Creation Order بالقائمة Edit (بالقائمة المختصرة بالفورمة)

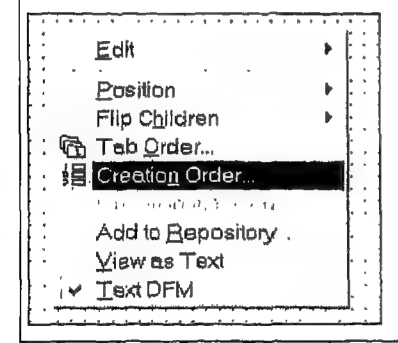
اختيار الأمر Creation Order من القائمة Edit يؤدى إلى فتح صندوق الحوار Creation Order الموضح فى الشكل التالى :


شكل توضيحى :





يمكنك أيضا الوصول إلى صندوق الحوار Creation Order عن طريق الأمر Creation Order الموجود بالقائمة المختصرة التى تظهر على الشاشة عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :

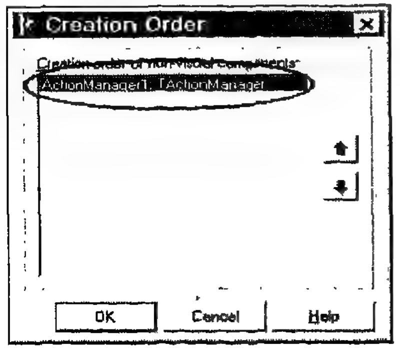


صندوق الحوار Creation Order

يمكـن اسـتخدام صـندوق الحوار Creation Order لتحديد ترتيب قيام التطبيق الذى تـتولى إعـداده بإنشـاء المكونات الغير مرئية وذلك عندما تقوم بتحميل الفورمة التى تشتمل على هذه النوعية من المكونات سواء فى مرحلة التصميم أو فى مرحلة التشغيل Run-Time.

قائمـة العرض Creation order of non-visual components تكون مشتملة فقط المكونـات الغـير مرئـية الموجـودة فـى الفورمـة النشـطة وهـى تعرض نوع كل مكون وترتيب الإنشاء الحالى له كما هو موضح بالشكل التالى :

شكل توضيحى :



فـى هـذا الصـدد نقول إن ترتيب الإنشاء الإفتراضى يتم تحديده عن طريق الترتيب الذى تم به وضع المكونات الغير مرئية فى الفورمة.

لكى تقوم بتغيير ترتيب الإنشاء لأى مكون من المكونات الغير مرئية اتبع الخطوات التالية :

(١) اخـتر اسـم المكـون الـذى ترغـبه فـى قائمة العرض Creation order of non-visual components كما هو موضح بالشكل التالى :

شكل توضيحى :

Creation Order
Creation order of non-visual components:
ActionManager1: TActionManager
CustomizeDlg1: TCustomizeDlg
OK Cancel Help

(٢) انقر بالماوس على المفتاح Up لنقل المكون المختار لأعلى فى قائمة العرض Creation order of non-visual components أو انقر بالماوس على المفتاح Down لنقله إلى أسفل فى قائمة العرض.



تستطيع أيضا سحب المكون المختار حتى تصل به إلى الموضع الذى ترغبه فى قائمة العرض Creation order of non-visual components.

(٣) لحفظ التغييرات التى قمت به انقر بالماوس على المفتاح Ok.

الأمر Flip Children بالقائمة Edit (بالقائمة المختصرة بالفورمة)

الأمر Flip Children الموجود فى القائمة Edit يسمح لك بأن تقلب أو تعكس تخطيط المكونات فى الفورمة الحالية لتصبح كما لو كنا ننظر إليها فى مرآة. ومثل هذا العمل يسمح للمبرمجين بأن يقوموا سريعا بتغيير أى فورمة من الوضع الطبيعى (من اليسار إلى اليمين) إلى الوضع المقابل أو المعاكس بحيث تصبح طبيعية بالنسبة لمستخدمى التطبيقات فى الوطن العربى.



يمكن الوصول لهذا الأمر عن طريق القائمة المختصرة التى تظهر عند النقر بالزر الأيمن للماوس على أى موضع بالفورمة كما هو موضح بالشكل التالى :

Edit
Position
Flip Children
All
Selected
Tab Order...
Creation Order...
Revert to Inherited
Add to Repository...
View as Text
Text DFM
Button1

الأمر All بقائمة الأمر Flip Children بالقائمة Edit

يعمل هذا الأمر على قلب أو عكس مواضع كافة العناصر الموجودة فى لفورمة. كذلك فإنه يقلب اتجاه ضبط أى أدوات تحكم تم ضبطها إلى يسار أو إلى يمين الفورمة.

الأمر Selected بقائمة الأمر Flip Children بالقائمة Edit

يعمل هذا الأمر على قلب أو عكس مواضع كافة العناصر الوليدة children الموجودة فى المكونات. كما إنه يعمل أيضا على عكس اتجاه ضبط أى مكونات قد تم ضبطها إلى يمين أو يسار المكونات المختارة.

الأمر Lock Controls بالقائمة Edit

اختيار الأمر Lock Controls من القائمة Edit يؤدى إلى تأمين كافة المكونات الموجودة فى الفورمة النشطة مما يجعلها ثابتة فى مواضعها ومن ثم لا يمكن نقلها إلى مواضع أخرى بالفورمة. هذا وعندما يكون هذا الأمر معلم عليه فإنك لن تستطيع نقل أو تغيير حجم أى أداة من أدوات التحكم ولكن فى نفس الوقت تستطيع أن تستخدم النافذة Object Inspector.



الخصائص التى تستطيع التعامل معها من خلال النافذة Object Inspector لأدوات التحكم فى هذه الحالة عبارة عن الخاصية Height والخاصية Left والخاصية Top والخاصية Width.

عندما يكون الأمر Lock Controls معلم عليه فى القائمة Edit تكون أدوات التحكم مقفلة locked وعندما تكون أدوات التحكم مقفلة تستطيع حينئذ أن تختار مرة أخرى الأمر Lock Controls من القائمة Edit لإلغاء التعليم من عليه ومن ثم تصبح أدوات التحكم غير مقفلة.



الأمر Lock Controls ليس له أى تأثير على الفورمة نفسها. وعندما تختر الأمر Lock Controls فستظل لديك القدرة على تغيير حجم الفورمة أو نقلها لأى موضع ترغبه بالشاشة.

الأمر Add to Interface بالقائمة Edit

اختـيار الأمر Add to Interface من القائمة Edit يعمل على تعريف إجراء جديد أو دالـة جديـدة أو خاصـية جديـدة لأى مكـون مـن الـنوع ActiveX. وهـذه العناصـر يـتم إضـافتها إلى واجهـات الاسـتخدام interfaces الخاصـة بـالمكون ActiveX ممـا يجعـلهم مـتاحين للاسـتخدام بالنسـبة للتطبـيقات الأخرى. وهذا الأمر عبارة عن اختصار shortcut لعملية الإعلان عن عضو وسيط لكى يتم استخدامه بواسطة أى مكون من النوع ActiveX.



تستطيع أيضا النقر بالزر الأيمن للماوس على ملف التنفيذ الخاصة بالمكون الـActiveX ثم تختار الأمر Add to Interface من القائمة المختصرة التى تظهر على الشاشة.

يعمل الأمر Add to Interface بالقائمة Edit على عرض صندوق الحوار Add to Interface الموضح فى الشكل التالى :

شكل توضيحى :

Add To Interface

Interface: Properties/Methods - IActiveFormX

Declaration:

Syntax Helper | OK | Cancel | Help

صـندوق الحـوار هـذا يسمح لك بأن تختار نوعية العضو الوسيط (فقد يكون خاصية أو أسلوب أو حدث) وبعد ذلك تقوم سريعا بإدخال إعلان declaration لهذا النوع المختار.

لعمل مراجعة تلقائية لصيغة ما تكتبة فى الحقل declaration فى هذه الحالة علم بالماوس على الاختيار Syntax Helper.

عـندما تـنقر بالماوس على المفتاح Ok يتم على الفور وبشكل تلقائى تخزين الإعلان الذى كتبته داخل المواضع الأساسية التالية :

- وحدة التنفيذ الحالية ActiveX Implementation.
- مكتبة الأنواع ActiveX (المخزنة فى ملف لة الامتداد TLB).
- ملف الاستيراد Pascal importer (عبارة عن الملف TLB.pas_).

والآن يمكنك بكل بساطة كتابة كود برمجي بلغة الباسكال لهذا لأسلوب أو تحدد قيمة لهذه الخاصية وذلك فى وحدة التنفيذ التى تتعامل معها.

القائمة Search

يمكن استخدام الأوامر الموجودة بالقائمة Search لكى تعثر على نصوص أو أخطاء أو كائنات أو وحدات أو متغيرات أو حتى رموز فى محرر الكود البرمجي Code Editor.

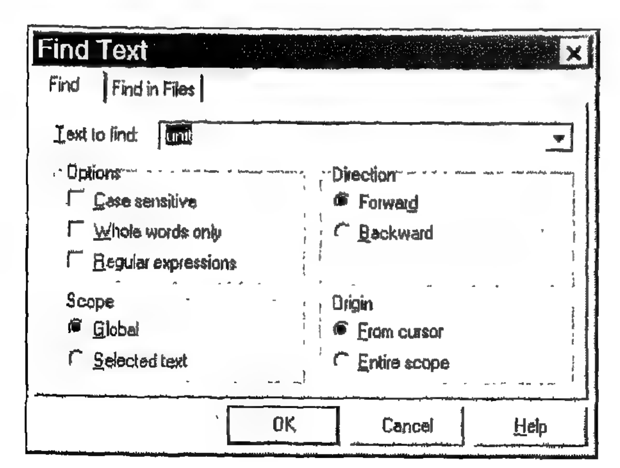
هذا والجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة بهذه القائمة :

الأمر	الوصف والاستخدام
Find	البحث عن نص معين والتعليم على أول نص يتم العثور عليه فى محرر الكود البرمجى Code Editor.
Find in Files	البحث عن نص معين وعرض كل نص يتم العصور عليه فى نافذة تقع بالجزء السفلى بمحرر الكود البرمجى Code Editor.
Replace	البحث عن نص معين واستبداله بنص جديد.
Search Again	إعادة آخر عملية بحث.
Incremental Search	البحث عن النص فى أثناء كتابته.
Go to Line Number	نقل مؤشر الكتابة لسطر معين فى محرر الكود البرمجى Code Editor.
Find Error	البحث عن آخر خطأ وقع فى مرحلة التشغيل Run-Time.
Browse Symbol	البحث عن رمز معين.

الأمر Find بالقائمة Search

اختيار الأمر Find من القائمة Search يؤدى إلى فتح صندوق الحوار Find Text الموضح فى الشكل التالى :

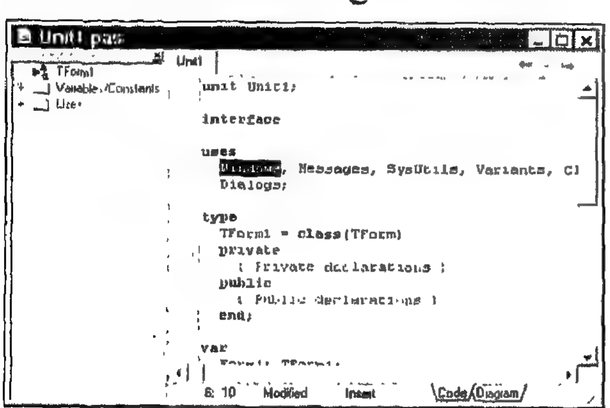
شكل توضيحى :



صندوق الحوار Find Text

من خلال التبويب Find بصندوق الحوار Find Text تستطيع أن تحدد النص الذى ترغب فى العثور عليه كما تستطيع أيضا من خلال هذا التبويب تحديد الخيارات التى تؤثر فى عملية البحث. هذا وصندوق الحوار Find Text يحدد موضع السطر بالكود البرمجى الذى يشتمل على أول نص يتم العثور عليه كما يقوم بالتعليم على هذا النص فى نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :



الخيارات الخاصة بصندوق الحوار Find Text

الحقل Text to find

ادخل السلسلة الحرفية المطلوب العثور عليها فى حقل القائمة المنسدلة Text to find او انقر بالماوس على السهم الموجود بهذه القائمة المنسدلة لفتحها ثم اختر نص من النصوص التى سبق البحث عنها كما هو موضح بالشكل التالى :

شكل توضيحى :

فيما يلى سنستعرض سويا الخيارات التى تحدد صفات عملية البحث والتى توجد فى القسم Options بصندوق الحوار :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون خذا الاختيار off-أى لا يعمل-يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الإعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.

فيما يلى سنستعرض سويا الخيارات التى تحدد اتجاه البحث :

الاختيار	الوصف والاستخدام
Forward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى نهاية الملف.
Backward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى بداية الملف.

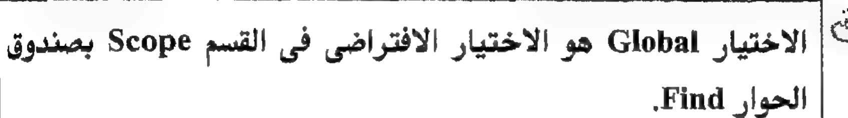


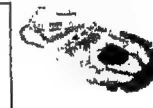
الاختيار Forward هو الاختيار الافتراضى.

فيما يلى سنستعرض سويا الاختيارات التى تحدد نطاق البحث داخل الملف :

الاختيار	الوصف والاستخدام
Global	هذا الاختيار يجعل نطاق البحث عبارة عن الملف بأكملة وذلك فى الاتجاه المحدد بواسطة الخيارات التى تحدد إتجاه البحث.
Selected text	من خلال هذا الأمر يتم البحث فى النص المختار بالإتجاه المحدد بالخيارات التى تحدد إتجاه البحث. وأنت تستطيع استخدام الماوس أو أوامر البلوك لاختيار بلوك نصى.
From cursor	هذا الاختيار يجعل عملية البحث تبدأ من عند الموضع الحالى لمؤشر الكتابة وبعد ذلك تستمر إما للأمام حتى نهاية مدى البحث أو للخلف حتى بداية مدى البحث وهذا يتحدد بناء على الاختيارات التى تحدد اتجاه البحث.
Entire Scope	من خلال هذا الاختيار تغطى عملية البحث إما البلوك كله الموجود به النص المختار أو تغطى الملف بأكمله (لا يهم أين يوجد مؤشر الكتابة فى الملف) وهذا يتحدد بناء على الخيارات الموجودة فى القسم Scope بصندوق الحوار Find.




الاختيار Global هو الاختيار الافتراضى فى القسم Scope بصندوق الحوار Find.



الاختيار From cursor هو الاختيار الافتراضى بالقسم Origin فى صندوق الحوار Find.

الأمر Find in Files بالقائمة Search

اختيار الأمر Find in Files من القائمة Search يؤدى إلى فتح صندوق الحوار Find Text الموضح فى الشكل التالى :

شكل توضيحى :

Find Text
Find in Files
Text to find: var
Options
Case sensitive
Whole words only
Regular expressions
Where
Search all files in project
Search all open files
Search in directories
Search Directory Options
File mask:
OK　Cancel　Help

صندوق الحوار Find Text

صندوق الحوار Find Text يشتمل على تبويب واحد فقط وهو التبويب Find in Files وفى هذا التبويب يتم تحديد النص الذى ترغب فى تحديد موضعه كما يتم فيه تحديد الخيارات التى تؤثر فى عملية البحث. هذا وكل وجود للنص (السلسلة الحرفية) يتم سرده فى صندوق يوجد فى أسفل نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

```
Unit1.pas
TForm1
Variables/Constants
Uses
Unit1

unit Unit1;

interface

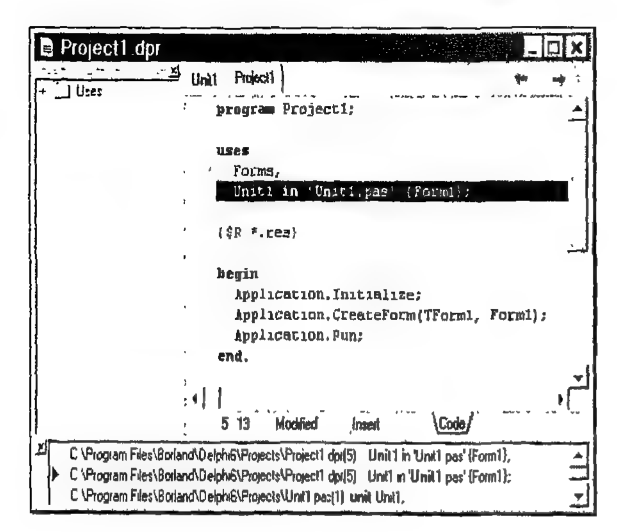
uses
  Windows, Messages, SysUtils, Variants, Cl
  Dialogs;

type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }

1: 1   Modified   Insert   Code  Diagram
C:\Program Files\Borland\Delphi6\Projects\Project1.dpr(5): Unit1 in 'Unit1.pas' {Form1};
C:\Program Files\Borland\Delphi6\Projects\Project1.dpr(5): Unit1 in 'Unit1.pas' {Form1};
C:\Program Files\Borland\Delphi6\Projects\Unit1.pas(1): unit Unit1;
```

وعـن طريق النقر بالماوس نقرا مزدوجا على أى عنصر من العناصر الموجودة فى هذا الصـندوق يتم الإنتقال إلى السطر الموجود به هذا العنصر بملف الكود البرمجى كما هو موضح بالشكل التالى :

شكل توضيحى :



حقل القائمة المنسدلة Text to Find

فـى حقـل القائمة المنسدلة Text to Find تكتب السلسلة الحرفية التى ترغب فى البحث عنه. ولكى تختار من بين السلاسل الحرفية التى سبق البحث عنها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :

شكل توضيحى :

Find Text
Find in Files
Text to find: Unit1
unit
Var
Public
Windows
Options
Case sen
Whole w
Regular expressions
Search in directories
Search Directory Options
File mask:
OK Cancel Help

الخيارات الموجودة فى القسم Options

الجدول التالى يقدم لنا وصف مختصر لوظيفة كل اختيار من الخيارات الموجودة فى القسم Options :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون خذا الاختيار off–أى لا يعمل–يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الإعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.

فيما يلى سنستعرض سويا الخيارات الموجودة فى القسم Where بصندوق الحوار Find in Files وذلك من خلال الجدول التالى :

الاختيار	الوصف والاستخدام
Search all files in project	مـن خـلال هـذا الاختـيار يتم البحث فى كافة الملفات الموجودة فى المشروع المفتوح حاليا.
Search all open files	مـن خلال هذا الاختيار يتم البحث فى كافة الملفات المفتوحة حاليا فقط.
Search in directories	عـند التعلـيم عـلى هذا الاختيار تصبح الخيارات الموجودة فى القسم Search Directory Options مـتاحة للاسـتخدام كمـا هـو موضـح بالشكل التالى : Find Text Find in Files Text to find: Options: Case sensitive / Whole words only / Regular expressions Where: Search all files in project / Search all open files / Search in directories Search Directory Options File mask: Browse... Include subdirectories OK Cancel Help وفـى هـذه الحالـة تـتم عملية البحـث عـبر كافـة الملفات الموجودة بالفهرس الذى يتم تحديده فى القائمة المنسدلة File mask.

الاختيارات الموجودة بالقسم Search Directory Options

مـن خـلال الجـدول الـتالى نسـتعرض سويا الخيارات الموجودة فى القسم Search Directory Options :

الاختيار	الوصف والاستخدام
File mask	من خلال هذه القائمة المنسدلة يتم تحديد المسار الدال على مواقع الملفات التي سيتم البحث داخلها. ومن الطبيعي أن يتم البحث في الملفات التي لها الامتداد PAS. والإمتداد DPR. فقط. أما لكي يتم البحث في ملفات أخرى في هذه الحالة استخدم حروف البحث التجميعية (مثل *.* أو txt.*) في نهاية المسار. لكي يتم البحث عن ملفات موجودة في الفهرس الأصلي الذى تم فيه تركيب لغة Delphi في هذه الحالة استخدم $(DELPHI)$ لتحديد هذا الفهرس. فعلى سبيل المثال للبحث بكافة الملفات الموجودة في الفهرس Include في هذه الحالة استخدم $(DELPHI)\Include أما لكي يتم البحث بالملفات الموجودة في المجلدات الموجود بها أمثلة في هذه الحالة استخدم $(DELPHI)\demos*.pas وهكذا... كذلك تستطيع أن تستخدم المفتاح Browse للتحديد المسار الذى ترغبه وذلك من خلال صندوق الحوار Browse for folders الموضح في الشكل التالى : Browse for Folder Select folder to search: Desktop My Computer 3.5 Floppy (A:) Sherif (C:) Sherif (D:) Sherif (E:) (F:) (G:) My Documents Network Neighborhood My Briefcase Online Services Cancel



Include subdirectories	لـو أنـك علمـت بالمـاوس عـلى هـذا الاختـيار فـإن ذلك سيؤدى إلى البحـث بالملفـات الموجـودة بـالمجلدات الفرعـية الموجـودة بـالمجلد المسار إليه بالمسار.



إذا أردت أن تدخل أكثر من مسار بحث فى القائمة المنسدلة File mask فى هذه الحالة يجب أن تفصل بين كل مسار والذى يليه بالفاصلة المنقوطة.

الأمر Replace بالقائمة Search

اختـيار الأمـر Replace مـن القائمـة Search يعمـل عـلى عـرض صـندوق الحوار Replace Text الموضح فى الشكل التالى :

شكل توضيحى :

Replace Text
Text to find: Unit1
Replace with:
Options:
Case sensitive
Whole words only
Regular expressions
Prompt on replace
Direction:
Forward
Backward
Scope:
Global
Selected text
Origin:
From cursor
Entire scope
OK | Replace All | Cancel | Help

صندوق الحوار Replace Text

يمكـن اسـتخدام صـندوق الحـوار Replace Text لـتحديد الـنص الذى ترغب فى البحث عنه لاستبداله بنص آخر (أو بلا شيء).

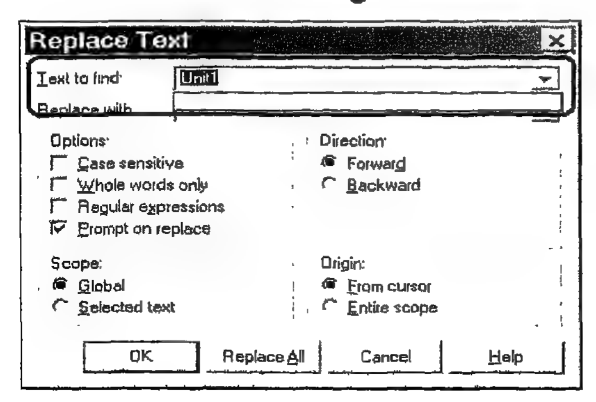


أغلب المكونات الموجودة فى صندوق الحوار Replace Text تتطابق مع المكونات الموجودة فى صندوق الحوار Find Text.

حقل القائمة المنسدلة Text to Find

فى حقل القائمة المنسدلة Text to Find تكتب السلسلة الحرفية التى ترغب فى البحث عنه. ولكى تختار من بين السلاسل الحرفية التى سبق البحث عنها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :

شكل توضيحى :



حقل القائمة المنسدلة Replace With

فى حقل القائمة المنسدلة Replace With تكتب السلسلة الحرفية التى سيتم استبدالها بالنص الموجود فى حقل القائمة المنسدلة Text to Find. ولكى تختار من بين السلاسل الحرفية التى سبق إدخالها فى هذه الحالة افتح هذه القائمة المنسدلة واختر منها ما ترغبه كما هو موضح بالشكل التالى :

شكل توضيحى :

الخيارات الموجودة فى القسم Options

الجدول التالى يقدم لنا وصف مختصر لوظيفة كل اختيار من الخيارات الموجودة فى القسم Options :

الاختيار	الوصف والاستخدام
Case sensitive	تمييز الحروف الكبيرة مثل ABC عن الحروف الصغيرة مثل a,b,c عند أداء عملية البحث.
Whole words only	البحث عن كلمات فقط. (ومن خلال كون خذا الاختيار off-أى لا يعمل-يتم البحث عن السلسلة الحرفية داخل سلاسل حرفية أكبر.
Regular expressions	الأخذ فى الإعتبار التعبيرات العادية فى أثناء البحث عن السلسلة الحرفية.
Prompt on replace	التعليم على هذا الاختيار يؤدى إلى ظهور الرسالة الموضحة فى الشكل التالى وذلك قبل إجراء عملية الاستبدال : Confirm Replace this occurrence of 'Unit1'? Yes No Cancel All وهذه الرسالة تطلب منك تأكيد لعملية الاستبدال. ولكن فى حالة عدم التعليم على هذا الاختيار فستتم عملية الاستبدال بدون أن تطلب منك تأكيد.

فيما يلى سنستعرض سويا الخيارات الموجودة فى القسم Direction والتى تحدد اتجاه البحث :

الاختيار	الوصف والاستخدام
Forward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى نهاية الملف.
Backward	جعل البحث يتم بداية من الموضع الموجود به مؤشر الكتابة حتى بداية الملف.

فيما يلى سنستعرض سويا الاختيارات الموجودة فى القسم Scope والتى تحدد نطاق البحث داخل الملف :

الاختيار	الوصف والاستخدام
Global	هذا الاختيار يجعل نطاق البحث عبارة عن الملف بأكمله وذلك فى الاتجاه المحدد بواسطة الخيارات التى تحدد إتجاه البحث.
Selected text	من خلال هذا الأمر يتم البحث فى النص المختار بالإتجاه المحدد بالخيارات التى تحدد إتجاه البحث. وأنت تستطيع استخدام الماوس أو أوامر البلوك لاختيار بلوك نصى.

فيما يلى سنستعرض سويا الخيارات الموجودة فى القسم Origin والتى تحدد نقطة البداية :

الاختيار	الوصف والاستخدام
From cursor	هذا الاختيار يجعل عملية البحث تبدأ من الموضع الحالى لمؤشر الكتابة ثم تنطلق إما للأمام حتى الوصول لنهاية مدى أو نطاق البحث أو للخلف حتى الوصول لبداية مدى أو نطاق البحث.
Entire scope	من خلال هذا الاختيار يتم البحث فى كل البلوك الموجود به النص المختار أو فى الملف بأكمله (بغض النظر عن موضع مؤشر الكتابة بالملف) وذلك اعتمادا على الخيارات الموجودة فى القسم Scope.

المفتاح Replace All

النقر بالماوس على المفتاح Replace All يؤدى إلى استبدال كافة النصوص التى يتم العثور عليها (المطابقة للنص الموجود فى حقل القائمة المنسدلة Text to find) بالنص الموجود فى حقل القائمة المنسدلة Replace with. ولو كنت قد علمت على الاختيار Prompt on replace فسيؤدى ذلك إلى ظهور رسالة التأكيد Confirm قبل إجراء أى استبدال.

الأمر Search Again بالقائمة Search

اختيار الأمر Search Again من القائمة Search يعمل على تكرار آخر عملية بحث أو استبدال. ونود هنا القول بأن القيم التحديدية التى تم إعدادها مؤخرا بصندوق الحوار Find Text أو بصندوق الحوار Replace Text تبقى كما هى وتؤثر نفس التأثير عندما تختار الأمر Search Again.

فعلى سبيل المثال لو أنك لم تقم بإلغاء التعليم من على الخيارات الموجودة فى صندوق الحوار Replace Text فى هذه الحالة سيقوم الأمر Search Again بالبحث عن آخر سلسلة حرفية ادخلتها فى حقل القائمة المنسدلة Text to find ثم يستبدلها بالسلسلة الحرفية الموجودة فى حقل القائمة المنسدلة Replace With.

الأمر Incremental Search بالقائمة Search

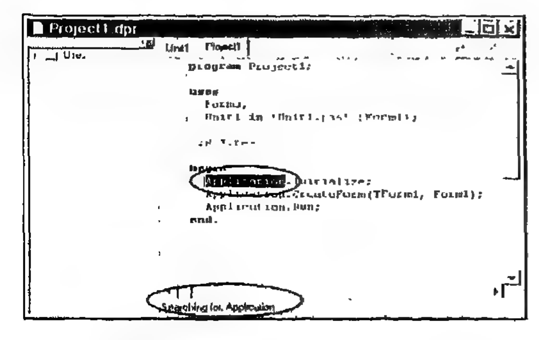
اختيار الأمر Incremental Search من القائمة Search يؤدى إلى تخطى bypass صندوق الحوار Find Text وذلك عن طريق نقل مؤشر الكتابة مباشرة إلى بداية النص الذى تكتبه بلوحة المفاتيح.

عندما تؤدى عملية بحث تزايدية ستلاحظ أن شريط الحالة لنافذة محرر الكود البرمجى Code Editor يكون مشتملا على الجملة Searching for كما إنه يعرض كل حرف تكتبه كما هو موضح بالشكل التالى :

شكل توضيحى :

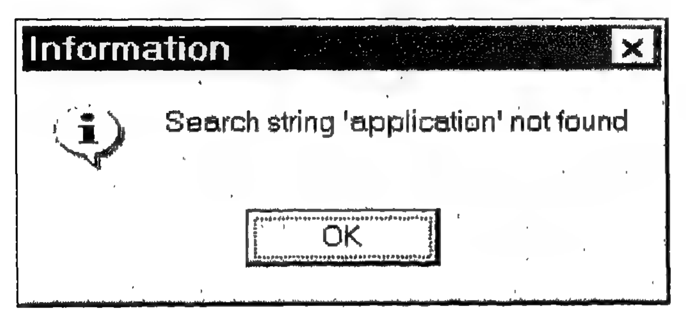
فعلى سبيل المثال لو أنك تكتب الكلمة Application ستجد أن مؤشر الماوس قد تحرك أى أول كلمة Application يتم العثور عليه كما يتم التعليم على كل حرف فيها فى أثناء الكتابة كما هو موضح بالشكل التالى :

شكل توضيحى :



ويستمر هذا السلوك حتى لا يكون هناك أى إحتمال للعثور على الكلمة المراد البحث عنا وفى هذه الحالة تظهر الرسالة الموضحة فى الشكل التالى :

شكل توضيحى :



فيما يلى سنستعرض سويا من خلال الجدول التالى استخدامات بعض المفاتيح بلوحة المفاتيح فى أثناء إجراء عملية البحث التزايدية :



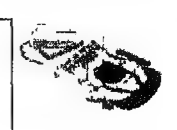
المفتاح	الاستخدام
Backspace	إزالة آخر حرف من السلسلة الحرفية والتحرك إلى آخر سلسلة حرفية تم العثور عليها سابقا.
F3	تكرار عملية البحث كما يمكن أيضا القيام بذلك من خلال الضغط على المفتاحين Ctrl+L بلوحة المفاتيح.

الأمر Go to Line Number بالقائمة Search

اختيار الأمر Go to Line Number من القائمة Search يؤدى إلى فتح صندوق الحوار Go To Line Number الموضح فى الشكل التالى :

شكل توضيحى :

Go to Line Number
Enter new line number: 12
OK Cancel Help



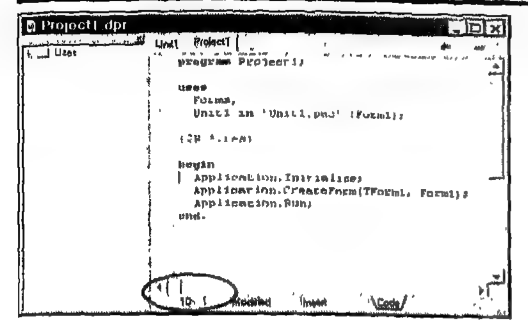
الضغط على المفتاحين Alt+G بلوحة المفاتيح يؤدى أيضا إلى فتح صندوق الحوار Go To Line Number.

صندوق الحوار Go To Line Number

صندوق الحوار Go To Line Number يطلب منك أن تدخل رقم السطر الذى ترغب فى العثور عليه بالكود البرمجى. هذا وستجد أن رقم السطر الموجود به مؤشر الكتابة حاليا معروضا فى صندوق الحوار Go To Line Number عند عرضه على الشاشة.

عند النقر بالماوس على المفتاح Ok يتم على الفور عرض كل من رقم السطر ورقم العمود الذى تم الوصول إليه وهذا العرض يكون فى شريط الحالة لنافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

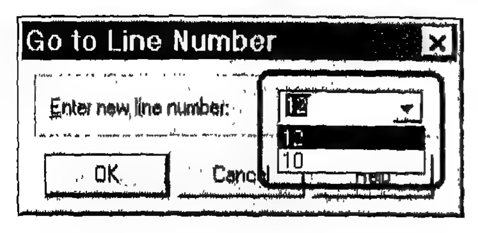
شكل توضيحى :



حقل القائمة المنسدلة Enter New Line Number

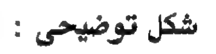
من خلال حقل القائمة المنسدلة Enter New Line Number يتم تحديد رقم السطر الذى ترغب فى وضع مؤشر الكتابة عنده فى نافذة محرر الكود البرمجى Code Editor. هذا ولكى تختار من أرقام السطور التى سبق إدخالها فى هذه الحالة افتح القائمة المنسدلة واختر منها الرقم الذى ترغبه كما هو موضح بالشكل التالى :

شكل توضيحى :

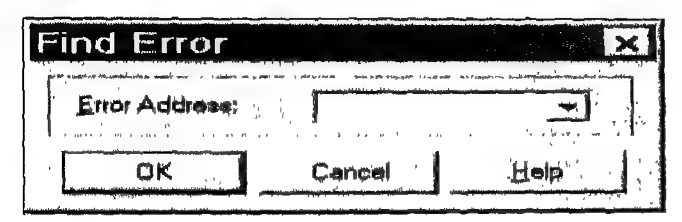


الأمر Find Error بالقائمة Search

اختيار الأمر Find Error من القائمة Search يؤدى إلى فتح صندوق الحوار Find Error الموضح فى الشكل التالى :



شكل توضيحى :





هذا الأمر يكون متاح للعمل فقط بعد أن يتم تشغيل التطبيق (وليس بعد أن يتم بناؤه فقط).

صندوق الحوار Find Error

يمكـن اسـتخدام صـندوق الحوار Find Error لتحديد عنوان أحدث خطأ وقع فى مرحلة التشغيل Run-Time.

حقل القائمة المنسدلة Error Address

كـل من عنوان أحدث خطأ تشغيلى ورقم الخطأ يظهران فى تقرير مرحلة التشغيل Run-Time لـو كانت هذه المعلومات متاحة. هذا وعندما تنقر بالماوس على المفتاح Ok تقوم لغـة Delphi بإعادة ترجمة البرنامج الذى تعده وتقف عند عنوان الخطأ الذى أدخلته ويتم التعليم على الخطأ كما هو موضح بالشكل التالى :

شكل توضيحى :

```
Unit1.pas
+ TForm1
+ Variables/Constants
+ Uses

end;

procedure TForm1.FileSaveExecute(Sender: TC
begin
        if (FileName = 'untitled.txt')  the
            FileSaveAs1.Execute
        else
            RichEdit1.Lines.SaveToFile(
end;

procedure TForm1.FileSaveAs1BeforeExecute(S
begin
        FileSaveAs1.Dialog.InitialDir := E
end;

procedure TForm1.FileSaveAs1Accept(Sender:
begin

72: 1        Insert        Code  Diagram
```

الأمر Browse Symbol بالقائمة Search

اختـيار الأمر Browse Symbol من القائمة Search يؤدى إلى فتح صندوق الحوار Browse Symbol الموضح فى الشكل التالى :

شكل توضيحى :

```
Browse Symbol
Symbol to find:
OK    Cancel    Help
```

صندوق الحوار Browse Symbol

يتم استخدام صندوق الحوار Browse Symbol لعرض رمز معين. هذا وللقيام بذلك يمكنك القيام بأى من الآتى :

- كتابة اسم الرمز فى حقل القائمة المنسدلة Symbol to find ثم النقر بالماوس على المفتاح Ok.
- فتح القائمة المنسدلة Symbol to find واختيار واحد من الرموز التى سبق التعامل معها بصندوق الحوار ثم النقر بالماوس على المفتاح Ok.

عند النقر بالماوس على المفتاح Ok تقوم لغة Delphi بعرض معلومات عن الرمز الموصف فى صندوق الحوار Browse Symbol وهذه المعلومات تظهر فى نافذة تسمى Symbol Explorer. وهذه النافذة تعمل على توفير معلومات عن الإشارات المرجعية الموجود بالكود البرمجى للرمز المختار كما هو موضح بالشكل التالى :

شكل توضيحى :

إذا كان الرمز عبارة عن قطاع class فى هذه الحالة نجد أن النافذة Symbol Explorer تعمل أيضا على توفير معلومات عن أعضاء هذا القطاع.



النافذة Symbol Explorer تشبه إلى حد كبير الجزء الأيمن للنافذة Project Browser كما هو موضح بالشكل التالى :

القائمة View

يمكن استخدام الأوامر الموجودة فى القائمة View لعرض أو إخفاء مختلف العناصر التى تتألف منها بيئة التطوير المتكاملة IDE للغة Delphi كما يمكن أيضا فتح النوافذ التى تنتمى إلى الأداة المتكاملة لمعالجة الأخطاء.

الجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة بالقائمة View :.

الأمر	الاستخدام
Project Manager	عرض مدير المشروع Project Manager على الشاشة.
Translation Manager	عرض مدير الترجمة Translation Manager على الشاشة
Object Inspector	عرض النافذة Object Inspector على الشاشة.
Object TreeView	استعراض الكائنات الموجودة فى فورمة من خلال مشهد شجرى.
To-Do List	هذا الأمر يسمح لك بأن تشاهد قائمة الواجبات الملحقة بالمشروع الذى يتم التعامل معها.
Alignment Palette	عرض بالليتة الضبط على الشاشة.
Browser	عرض نافذة مستعرض المشروع Project Browser.

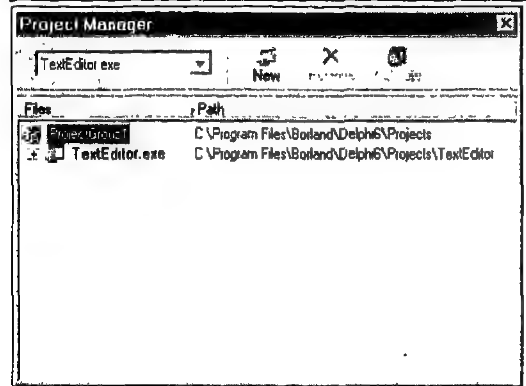
Code Explorer	عرض نافذة متصفح الكود البرمجى Code Explorer.
Component List	فتح صندوق حوار المكونات Components.
Window List	عرض قائمة تضم اسماء كافة النوافذ المفتوحة حاليا بالمشروع.
Debug Windows	عرض القائمة الفرعية التى تشتمل على الأوامر الخاصة بأداة اكتشاف الأخطاء ومعالجتها Debugger.
Desktops	هذا الأمر يسمح لك بأن تعرض أو حفظ أو مسح المشاهد المتعددة والمختلفة لسطح المكتب.
Toggle Form/Unit	من خلال هذا الأمر يتم التنقل من الفورمة إلى نافذة الوحدة والعكس.
Units	فتح صندوق الحوار View Unit.
Forms	فتح صندوق الحوار View Form.
Type Library	عرض نافذة محرر مكتبة النوع Type Library.
New Edit Window	فتح محرر كود برمجى Code Editor جديد.
Toolbars	إخفاء أو إظهار شرائط الأدوات أو بالليتة المكونات.

الأمر Project Manager بالقائمة View

اختيار الأمر Project Manager من القائمة View يؤدى إلى فتح نافذة مدير المشروع Project Manager الموضحة فى الشكل التالى :



شكل توضيحى :





إذا كانت نافذة مدير المشروع Project Manager مفتوحة بالفعل فى هذه الحالة تصبح النافذة النشطة.



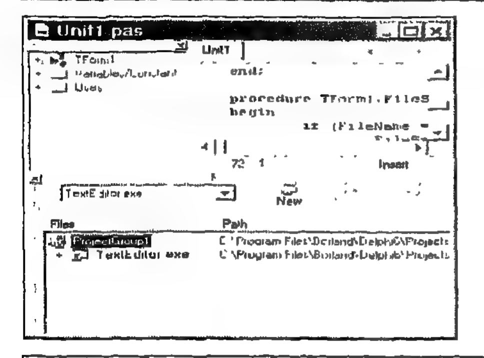
هناك طريقة أخرى لفتح نافذة مدير المشروع Project Manager وهى الضغط على مجموعة المفاتيح Ctrl+Alt+F11 بلوحة المفاتيح.

يمكن استخدام نافذة مدير المشروع Project Manager لمشاهدة مجموعة مشروع ومشاهدة المشروعات الموجودة فى مجموعة مشروع بالإضافة إلى إمكانية التجول بين الملفات الخاصة بالمشروع. وأنت تستطيع استخدام نافذة مدير المشروع Project Manager لإضافة مشاريع إلى مجموعة المشروع أو لمسح مشروعات منها أو لتنشيط (تفعيل) مشروع لو أن مجموعة المشروع التى تتعامل معها كانت تتألف من أكثر من مشروع.

كذلك تستطيع استخدام نافذة مدير المشروع Project Manager لإضافة أو مسح أو حفظ أو نسخ ملف إلى المشروع الذى يتم التعامل معه حاليا. وفى هذا الصدد نقول إن نافذة مدير المشروع Project Manager تعرض كافة الوحدات والفورمة المرتبطة بها والتى توجد فى المشروعات الموجودة بدورها داخل مجموعة المشروع التى يتم التعامل معها حاليا.

هذا وأنت تستطيع وضع نافذة مدير المشروع Project Manager فى أى مكان بسطح المكتب. كما إنك تستطيع أيضا جعلها راسية docking أو ملتصقة مع نوافذ أخرى مثل نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :





في هذا الشكل نشاهد نافذة مدير المشروع Project Manager وهى ملتصقة بالجزء السفلى بمحرر الكود البرمجى Code Editor.

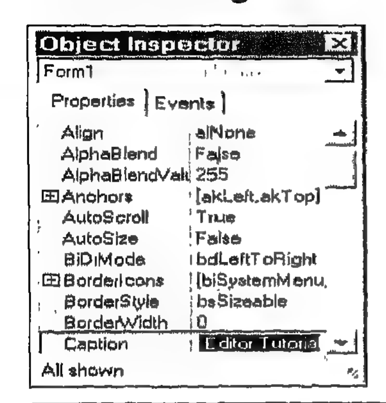
الأمر Translation Manager بالقائمة View

اختيار الأمر Translation Manager من القائمة View يؤدى إلى فتح نافذة مدير الترجمة Translation Manager. أما إذا كانت هذه النافذة مفتوحة بالفعل في هذه الحالة تصبح النافذة النشطة.

الأمر Object Inspector بالقائمة View

لو أنـك قمت بإغلاق النافذة Object Inspector فإن اختيار الأمر Object Inspector من القائمة View يؤدى إلى إعادة فتحها مرة أخرى وهذه النافذة نشاهدها فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح النافذة Object Inspector وهى الضغط على المفتاح F11 بلوحة المفاتيح.



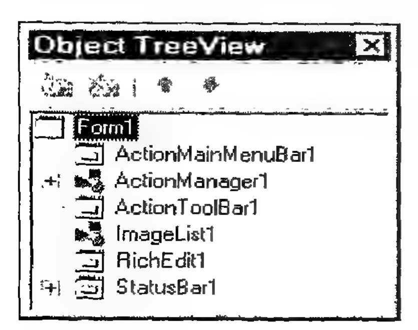
كذلك تستطيع اختيار الأمر Object Inspector من القائمة View للتنقل بين النافذة Object Inspector وآخر نافذة كانت نشطة أو نافذة محرر الكود البرمجى Code Editor.

هذا ويتم استخدام النافذة Object Inspector للتعديل فى قيم الخصائص والوصلات الخاصة بأدوات معاملة الأحداث.

الأمر Object TreeView بالقائمة View

اختيار الأمر Object TreeView من القائمة View يؤدى إلى عرض النافذة Object TreeView على الشاشة فوق النافذة Object Inspector والشكل التالى يوضح لنا النافذة Object TreeView :

شكل توضيحي :





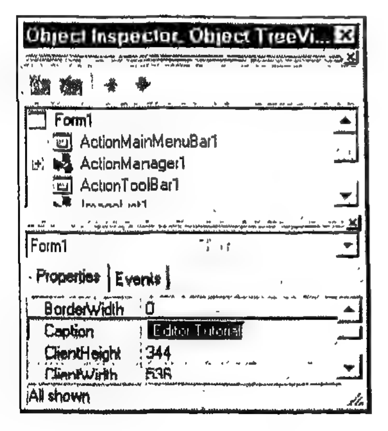
هناك طريقة أخرى لفتح النافذة Object Inspector وهى الضغط على مجموعة المفاتيح Shift+Alt+F11 بلوحة المفاتيح.

النافذة Object TreeView تعرض المكون المالك والمكونات الأخرى التابعة له (التى يطلق عليها أطفال). فعلى سبيل المثال تعتبر الفورمة هى المالك لأى مفتاح Button موجود داخلها وفى نفس الوقت نقول أن المفاتيح تعتبر أطفال الفورمة وهكذا...وفى هذا الصدد نقول إن هناك علاقة تبادلية مباشرة بين النافذة Object TreeView والنافذة Object Inspector ومن ثم فإن المشهد الشجرى يتغير عند حدوث أى تغيير فى النافذة Object Inspector والعكس صحيح أيضا.

تمتلك النافذة Object TreeView كافة القدرات والإمكانيات الخاصة بأداة تصميم Module البيانات. ومن بين هذه الإمكانيات إمكانية سحب واسقاط العناصر فى القوائم.

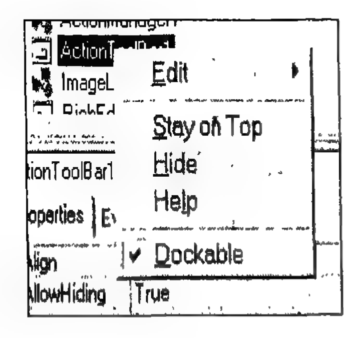
هـذا وتعتبر النافذة Object TreeView من النوافذ القابلة للإرساء dockable بحيث يمكن لصقها فى النافذة Object Inspector كما هو موضح بالشكل التالى :

شكل توضيحى :



النقر بالزر الأيمن للماوس على أى عنصر من العناصر المعروضة فى النافذة Object TreeView يـؤدى إلى ظهـور قائمـة مختصـرة تشـتمل على نفس الخيارات التى توجد فى القائمـة المختصـرة الـتى تظهـر عـند الـنقر بالـزر الأيمن للماوس داخل نافذة المحرر Data Module كما هو موضح بالشكل التالى :

شكل توضيحى :



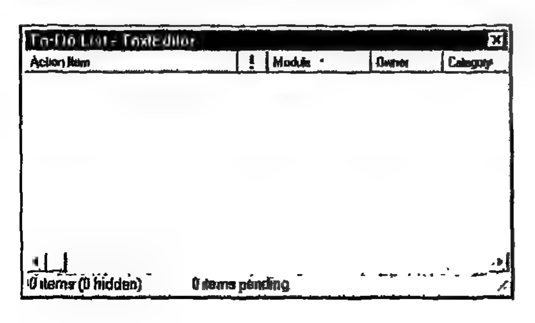
شـريط الأدوات الخـاص بالـنافذة Object TreeView تشـتمل عـلى كل من الأيقونة New Item والأيقونة Delete والأيقونة Move Up والأيقونة Move Down . ولـو أن هـناك أكـثر مـن نـوع من العناصر التى يمكن إضافتها فى هذه الحالة ستجد أن الأيقونـة New Item قد أصبح لها سهم صغير يشير لأسفل وعند النقر بالماوس على هذا السهم يؤدى إلى فتح قائمة صغيرة تشتمل على الأنواع المتاحة من العناصر التى يمكن إضافتها.

عندما يكون هناك كائن غير مكتمل فى هذه الحالة يتم عرضه فى النافذة Object TreeView وبجـواره نشاهد علامة صح حمراء. أما الأيقونات الشبحية Ghost فإنها تمثل مكونـات تم إنشـاؤها ضـمنيا لكـى يتم استخدامها فى الخلفية بدون أن يشعر بها المستخدم مثل المكون Default Session.

الأمر To-Do List بالقائمة View

اختـيار الأمـر To-Do List مـن القائمـة View يـؤدى إلى عرض النافذة To-Do List الموضحة فى الشكل التالى :

شكل توضيحى :.



وهـذه الـنافذة تكـون خاصـة بالمشـروع الـذى يتم التعامل معها حاليا. وهذه النافذة تعرض المهام التى ينبغى إنجازها لإستكمال المشروع الحالى.

العناصر الخاصة بالمشروع ككل يتم عرضها فى النافذة To-Do List. أما العناصر الموجودة فـى المشـروع والـتى تمتلك كود برمجى غير مفتوح فى محرر الكود البرمجى Code Editor فإنها تظهر باللون الرمادى فى هذه النافذة.

تسـتطيع أن تقـوم بفـرز وترتيـب العناصـر الموجودة فى النافذة To-Do List وهذا الترتيـب يمكـن أن يكون عن طريق الأسم أو عن طريق الحالة أو عن طريق الأهمية والأولوية وهذا الترتيب يتم عن طريق النقر بالماوس على رأس العمود المناسب.

الأمر Alignment Palette بالقائمة View

اختـيار الأمـر Alignment Palette مـن القائمـة View يـؤدى إلى عـرض بالليـتة الضبط الموضحة فى الشكل التالى :

شكل توضيحى :

Align

مـن خلال هذه الباللتة يمكنك ضبط مواضع المكونات على شبكة النقط الموجودة فى الفورمة أو ضبط كل مكون بالنسبة للمكونات الأخرى.



تستطيع أيضا ضبط مواضع المكونات عن طريق صندوق الحوار Alignment الموضح فى الشكل التالى :

Alignment

Horizontal
- No change
- Left sides
- Centers
- Right sides
- Space equally
- Center in window

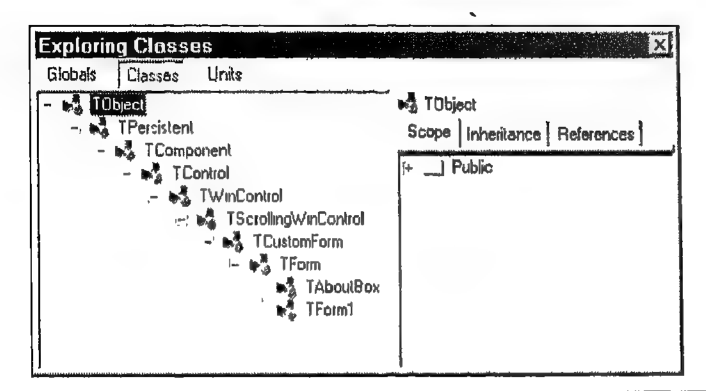
Vertical
- No change
- Tops
- Centers
- Bottoms
- Space equally
- Center in window

OK | Cancel | Help

الأمر Browser بالقائمة View

اختـيار الأمـر Browser من القائمة View يؤدى إلى فتح النافذة Project Browser الموضحة فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح النافذة Project Browser وهى الضغط على مجموعة المفاتيح Shift+Ctrl+B بلوحة المفاتيح.

هـذا ولكـى تتمكن من تحديد مدى أو مجال عمل النافذة Project Browser اتبع الخطوات التالية :

(١) افتح القائمة Tools ثم اختر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :

شكل توضيحى :

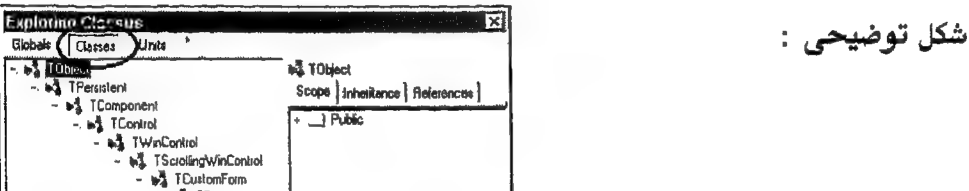
(٢) انقر بالماوس على التبويب Explorer ليظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :

(٣) بهـذا التبويب وفى القسم Browser Scope علم بالماوس إما على الاختيار Project symbols only أو على الاختيار All symbols الذى يتضمن المكتبة VCL ومـن ثم تستطيع إما أن تشاهد الرموز من المشروع الذى تعده فقط أو من كافة الوحدات التى يستخدمها المشروع الذى تعده.

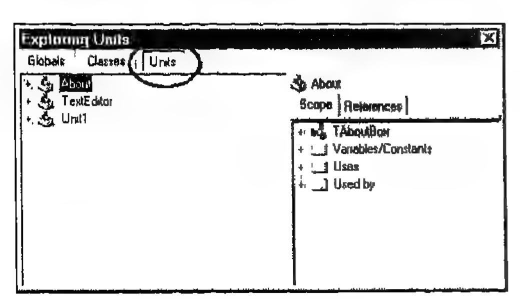
(٤) مـن خلال القسم Initial Browser view تستطيع أن تحدد المشهد الابتدائى للنافذة Project Browser بحيث يكون إما مشهد القطاعات classes الموضح فى الشكل التالى :

شكل توضيحى :



أو مشهد الوحدات Units الموضح فى الشكل التالى :

شكل توضيحى :



أو مشهد العموميات globals الموضح فى الشكل التالى :

شكل توضيحى :



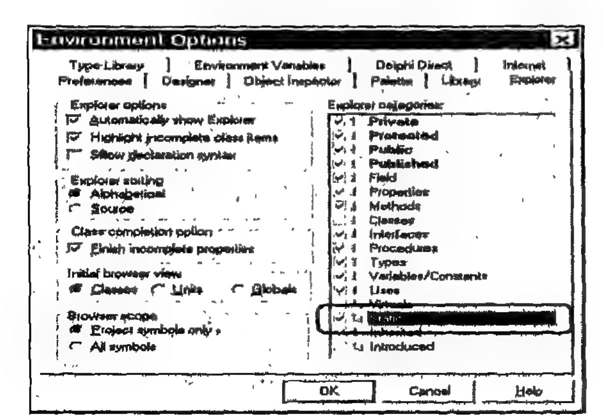


معنى ذلك أنه عندما تقوم بفتح النافذة Project Browser يظهر على الفور المشهد الإبتدائى بهذه النافذة.

الخـيارات الموجـودة فـى القسـم Explorer categories بالتبويـب Explorer بصـندوق الحـوار Environment Options تسـمح لك بأن تتحكم فى الطريقة التى يتم بها

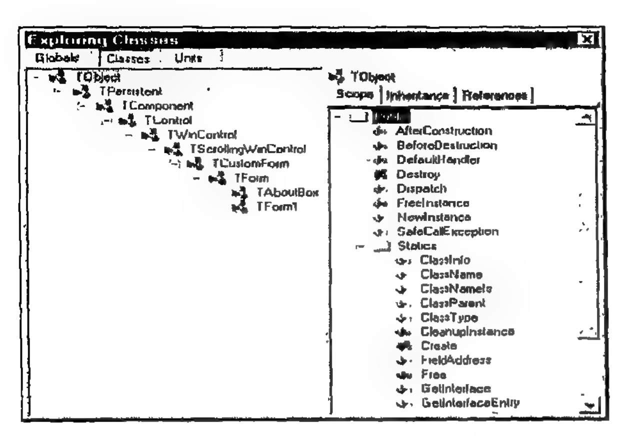
تجميع العناصر التى يتم عرضها داخل النافذة Project Browser. وفى حالة التعليم على أى اختيار من هذه الإختيارات تجد أنه ظهر عمود مناظر لهذا الاختيار فى النافذة Project Browser. فعلى سبيل المثال لو قمت بالتعليم على الاختيار Statics كما هو موضح بالشكل التالى :

شكل توضيحى :



فى هذه الحالة تصبح العناصر المعروضة فى النافذة Project Browser كما هو موضح بالشكل التالى :

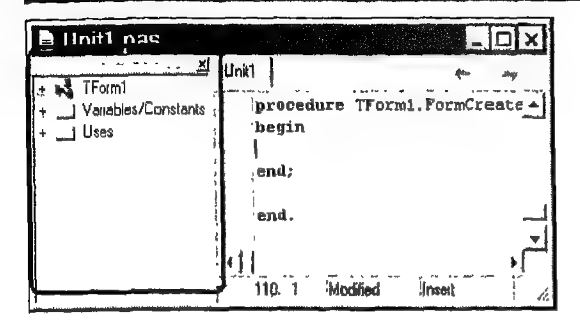
شكل توضيحى :



الأمر Code Explorer بالقائمة View

من الطبيعى أن تكون النافذة Code Explorer راسية أو ملتصقة بيسار محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

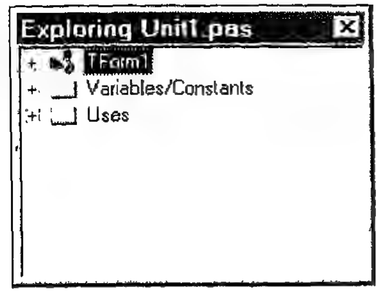


أما لو كانت هذه النافذة مغلقة تستطيع حينئذ إعادة فتحها عن طريق اختيار الأمر Code Explorer من القائمة View.

النافذة Code Explorer تجعل من السهولة بمكان التجول عبر ملفات الوحدات التى تتعامل معها بالإضافة إلى إمكانية إنشاء القطاعات تلقائيا.

- لكى تغلق النافذة Code Explorer فى هذه الحالة ينبغى جعلها منفصلة عن نافذة محرر الكود البرمجى Code Editor لتصبح كما هو موضح بالشكل التالى :

شكل توضيحى :



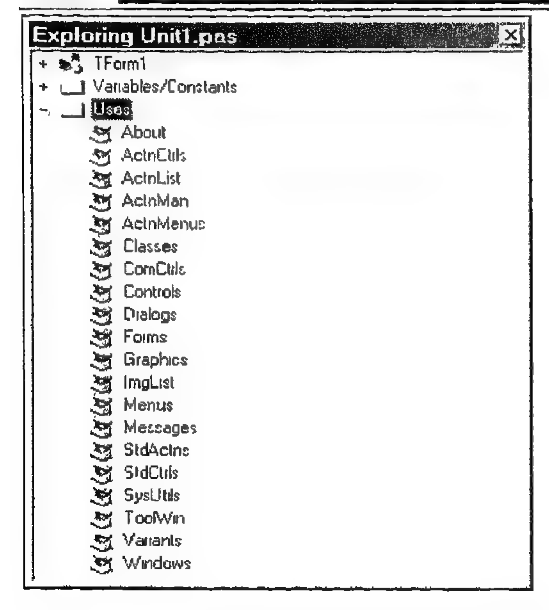
ثم تنقر بالماوس على الزر (×) بالركن الأيمن العلوى للنافذة.

- لكى تعيد فتح النافذة Code Explorer اختر الأمر Code Explorer من القائمة View.

النافذة Code Explorer تشتمل على ديجرام شجرى يوضح كافة أنواع القطاعات والخصائص والأساليب والمتغيرات العامة والروتينات العامة التى تم تعريفها فى ملف الوحدة الذى تتعامل معه. كما إنه يوضح أيضا الوحدات الأخرى من خلال المجلد uses كما هو موضح بالشكل التالى :



شكل توضيحى :



تستطيع فتح أو غلق المجلدات الموجودة فى المشهد الشجرى وذلك عن طريق العلامة (+) أو (–) الموجودة بجوار اسم المجلد الذى ترغب فى التعامل معه.

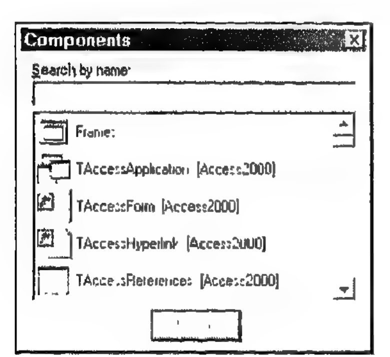


أى ملف وحدة يكون مفتوح فى نافذة محرر الكود البرمجي Code Editor فإنه يكون مفتوح أيضا فى النافذة Code Explorer.

الأمر Component List بالقائمة View

اختيار الأمر Component List من القائمة View يؤدى إلى عرض نافذة المكونات Components على الشاشة كما هو موضح بالشكل التالى :

شكل توضيحى :



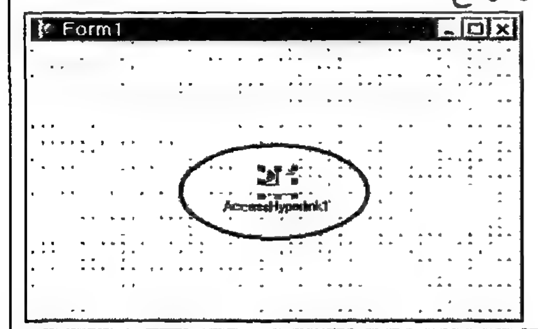
نافذة المكونات Components

يمكـن اسـتخدام نـافذة المكونـات Components لإضـافة مكونـات إلى الفورم التى تعدها بمشروعك وذلك باستخدام الماوس أو من خلال لوحة المفاتيح.

هـذا والجـدول الـتالى يقـدم لـنا وصـف مختصـر للعناصـر الـتى تـتألف منها نافذة المكونات Components :

العنصر	الوصف والاستخدام
الحقل Search by name	فـى هـذا الحقـل يـتم ادخـال اسـم المكـون الـذى ترغـب فـى إضافته للفورمـة. وهـذا الحقـل يـؤدى مهمـة بحث تزايدية ومن ثم تجد أن المؤشـر يـتحرك إلى المكـون الـذى يكـون اسمه مشتملا على الحروف التى تكتبها فى أثناء كتابتها بهذا الحقل.
قائمة العرض Component	عن طريق هذه قائمة العرض هذه يمكن أن تختار المكون الذى ترغب فـى إضافته. هذا وستجد أن المكونات معروضة فى هذه القائمة وهى مرتبة ترتيب أبجدى كما ستجد أن كل مكون يوجد بجواره الأيقونة التى تمثله.
الحقل Search	عـندما تخـتار مكـون فـإن اسمـه يظهـر فى الحقل Search كما هو موضح بالشكل التالى :



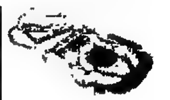
المفتاح Add to form	النقر بالماوس على المفتاح Add To Form يؤدى إلى وضع حالة instance للمكون المختار وستلاحظ أن المكون قد تم وضعه فى مركز الفورمة تماما كما هو موضح بالشكل التالى : 



لكى تختار Add To Form من لوحة المفاتيح اضغط على المفتاح Enter بلوحة المفاتيح.

لكى تضيف المكون الذى اخترته فى صندوق الحوار Component List عليك أن تقوم بواحد من الآتى :

- الضغط على المفتاح Enter بلوحة المفاتيح.
- النقر بالماوس نقرا مزدوجا على اسم المكون فى قائمة العرض Components.
- النقر بالماوس على اسم المكون فى قائمة العرض Components ثم النقر بالماوس على المفتاح Add To Form.



عندما تضيف مكون إلى فورمة عن طريق استخدام لوحة المفاتيح تجد أن لغة Delphi تستخدم الحجم الافتراضى للمكون كما إنها تضيف المكون فى مركز الفورمة إذا لم يكن هناك مكون حاوية (مثل صندوق التجميع أو اللوحة panel) مختارا فى الفورمة.

فى حالة وجود مكون حاوية مختار فى الفورمة فى هذه الحالة تقوم اللغة بوضع المكون الذى تضيفه فى مركز هذا المكون الحاوية. على العموم لكى تضيف مكون داخل حاوية فإنه ينبغى عليك اختيار الحاوية أولا قبل النقر بالماوس على المفتاح Add to form فى صندوق الحوار Component List.

الأمر Window List (بالقائمة View أو القائمة Window)

اختيار الأمر Window List من القائمة View أو من القائمة Window يؤدى إلى فتح صندوق الحوار Window List يضم قائمة بكافة النوافذ المفتوحة حاليا فى بيئة التطوير المتكاملة IDE. وهذا الشكل يوضح لنا صندوق الحوار هذا :

شكل توضيحى :

Window List
Unit1.pas
Text Editor Tutorial
Align
Object TreeView
Object Inspector
Delphi 6 - TextEditor
OK Cancel Help

عند اختيار اسم نافذة من النوافذ المعروضة فى صندوق الحوار Window List فإن ذلك يؤدى إلى جعل هذه النافذة نشطة أى يتم نقل دفة التحكم لها. ولو أن لديك الكثير من النوافذ مفتوحة فى هذه الحالة ستجد أن صندوق الحوار Window List يعتبر أسهل طريقة للذهاب إلى النافذة التى ترغبها.



هناك طريقة أخرى لفتح صندوق الحوار Window List وهى الضغط على المفتاحين Alt+0 بلوحة المفاتيح.

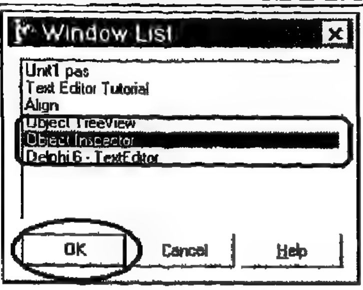
صندوق الحوار Window List

يمكن استخدام صندوق الحوار Window List لتحويل نافذة من كونها غير نشطة لتصبح نشطة. هذا ويقوم صندوق الحوار Window List باستعراض كافة النوافذ المفتوحة حاليا فى المشروع.

لكى تختار نافذة قم بأى من الآتى :

- النقر بالماوس نقرا مزدوجا على اسم النافذة.
- اختيار اسم النافذة ثم النقر بالماوس على المفتاح Ok كما هو موضح بالشكل التالى :

شكل توضيحي :



الأمر Debug Windows بالقائمة View

يمكن استخدام مجموعة الأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Debug Windows بالقائمة View لفتح النوافذ التى تنتمى إلى الأداة المتكاملة لاكتشاف الأخطاء ومعالجتها Integrated debugger. وهذه النوافذ توضح لنا كيف تتم عملية اكتشاف الثغرات التى توجد بالبرنامج ومعالجتها أيضا.

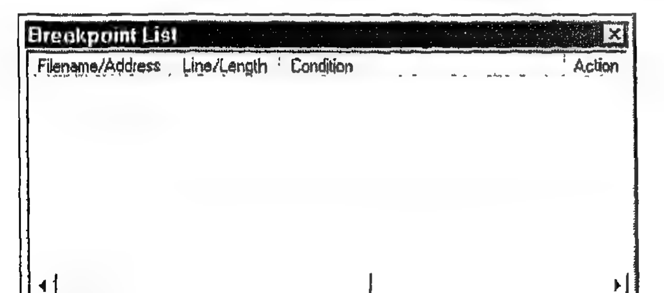
الجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Debug Windows بالقائمة View:

الأمر	الوصف والاستخدام
Breakpoints	يعمل هذا الأمر على عرض نافذة المراقبة Breakpoint List.
Call Stack	من خلال هذا الأمر يتم فتح نافذة المراقبة Call Stack.
Watches	عن طريق هذا الأمر يتم فتح نافذة المراقبة Watch List.
Local Variables	يعمل هذا الأمر على توضيح المتغيرات المحلية الخاصة بالدالة التى يتم تنفيذها حاليا فى أثناء كون البرنامج فة مود الـdebug.
Threads	من خلال هذا الأمر يتم فتح نافذة المراقبة Thread Status.
Modules	من خلال هذا الأمر يتم فتح نافذة المراقبة Modules.
Event Log	يعمل هذا الأمر يتم فتح نافذة المراقبة Event Log.
CPU	يعمل هذا الأمر يتم فتح نافذة المراقبة CPU.
FPU	يعمل هذا الأمر يتم فتح نافذة المراقبة FPU.

الأمر Breakpoints بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Breakpoints من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح النافذة Breakpoint List الموضحة فى الشكل التالى :


شكل توضيحى :





هناك طريقة أخرى لفتح النافذة Breakpoint List وهى الضغط على مجموعة المفاتيح Ctrl+Alt+B بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى النافذة Breakpoint List.

النافذة Breakpoint List تقوم بعرض كافة نقط التوقف المحددة حاليا. وكل نقطة توقف موجودة فى هذه النافذة نشاهد لها المعلومات الموضحة فى الجدول التالى :

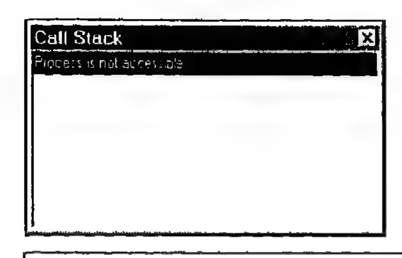
العمود	الوصف والاستخدام
Filename/Address	هذا العمود يعرض اسم الملف الذى تم فيه تحديد نقطة التوقف. كما يعمل أيضا على عرض اسم المتغير الذى تم فيه تحديد البيانات الخاصة بنقط التوقف. كذلك يمكن أن يعرض هذا العمود العناوين التى تم عندها تحديد نقط التوقف وذلك إذا لم يكن العنوان مرتبط بالسطر الذى تم فيه تحديد نقط التوقف وفى هذه الحالة يتم عرض اسم الملف الموجود به هذا السطر.
Line/Length	بالنسبة لـSource breakpoints نجد أن هذا العمود يعرض رقم

	السطر الموجود فيه نقطة التوقف. أما بالنسبة لـ Data breakpoints فإن هذا العمود يعرض طول نقطة التوقف. وبالنسبة للـ Address breakpoints يكون هذا العمود فارغ وذلك إذا لم تكن هناك إمكانية لربط العنوان بالسطر الموجود فيه لـSource breakpoints ولكن في حالة تحقق هذا الإرتباط فستجد أن هذا العمود يعرض رقم هذا السطر بالإضافة إلى الكود السداسى عشر للعنوان من بداية السطر.
Condition	هذا العمود يعرض أى حالة تكون مرتبطة بنقطة التوقف التى يجرى عرضها الآن.
Pass Count	هذا العمود يعرض أى رقم مرور مرتبط بنقطة التوقف التى يجرى عرضها الآن.

الأمر Call Stack بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Call Stack من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح النافذة Call Stack الموضحة فى الشكل التالى :


شكل توضيحى :





هناك طريقة أخرى لفتح النافذة Call Stack وهى الضغط على مجموعة المفاتيح Ctrl+Alt+S بلوحة المفاتيح.



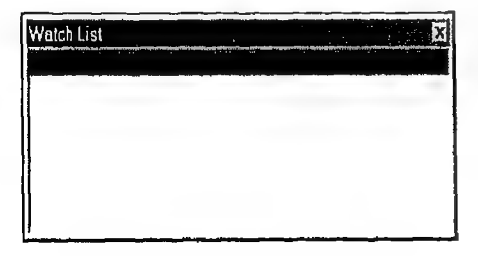
لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى النافذة Call Stack.

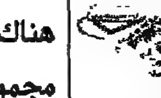
النافذة Call Stack تعرض التتابع الحالى للروتينات التى قام البرنامج باستدعاؤها فى مرحلة التشغيل Run-Time. وفى هذا العرض نجد أن آخر روتين تم استدعاؤه موجود فى القمة القائمة. هذا وكل عنصر فى النافذة Call Stack يعرض معلومات متمثلة فى كل من اسم الإجراء والقيم الخاصة بأى معاملات يتم تمريرها إليه.

الأمر Watches بالقائمة الفرعية Debug Windows من القائمة View

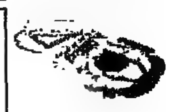
اختيار الأمر Watches من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح نافذة المراقبة Watch List الموضحة فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح نافذة المراقبة Watch List وهى الضغط على مجموعة المفاتيح Ctrl+Alt+W بلوحة المفاتيح.



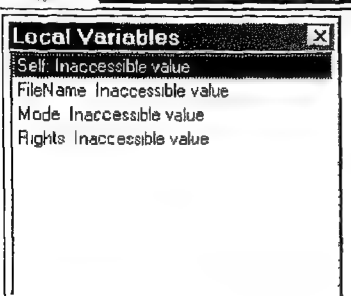
لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى نافذة المراقبة Watch List.

نافذة المراقبة Watch List تعرض كافة تعبيرات المراقبة التى تم تحديدها حاليا. ولو أنك احتفظت بهذه النافذة مفتوحة فى أثناء كون البرنامج فى مود الـDebug (مود اكتشاف الثغرات البرمجية ومعالجتها) فى هذه الحالة تستطيع أن تراقب سلوك البرنامج وهو يقوم بتحديث القيم المخصصة للمتغيرات الهامة فى أثناء تشغيل البرنامج.

الأمر Local Variables بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Local Variables من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى عرض القيم المخصصة حاليا للمتغيرات المحلية الخاصة بالدالة التى يتمك التعامل معها حاليا وذلك فى أثناء كون البرنامج فى مود الـdebug وذلك من خلال نافذة المراقبة Local Variables الموضحة فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح نافذة المراقبة Local Variables وهى الضغط على مجموعة المفاتيح Ctrl+Alt+L بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى نافذة المراقبة Local Variables.

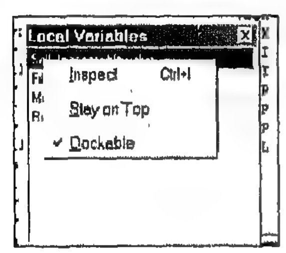
هـذا الأمر يكون دائما متاح ولكن نافذة المراقبة Local Variables تكون فارغة إذا لم تكـن أداة فحـص الأخطاء ومعالجتها debugger معلقة paused. هذا ولو أنك احتفظت بهـذه الـنافذة مفتوحة فـى أثـناء القـيام بمهام اكتشاف الأخطاء ومعالجتها فى هذه الحالة تسـتطيع تراقـب سـلوك الـبرنامج وهـو يقوم بتحديث القيم المخصصة للمتغيرات الـهامة فى أثناء تشغيل البرنامج..

القائمة المختصرة الخاصة بنافذة المراقبة Local Variables

عـند الـنقر بالـزر الأيمـن لـلماوس عـلى أى عنصـر مـن العناصر الموجودة فى نافذة المراقبة Local Variables تظهر على الشاشة القائمة المختصرة الموضحة فى الشكل التالى :



شكل توضيحى :



الجـدول التالى يقدم لنا وصف مختصر لوظيفة الأوامر الموجودة فى القائمة المختصرة الموضحة فى الشكل السابق.

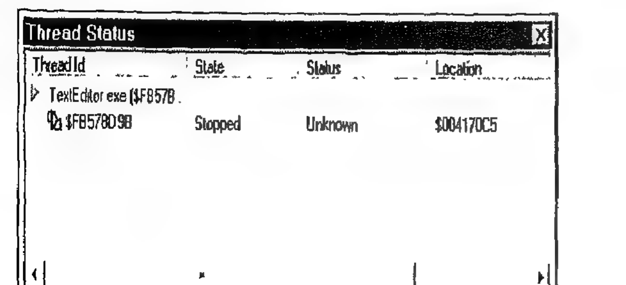
الأمر	الوصف والاستخدام
Inspect	يقوم هـذا الأمـر بعـرض معلومـات عـن المـتغير المخـتار حاليا وهذه المعلومـات يتم عرضها فى التبويب Data بالنافذة Debug Inspector كما هو موضح بالشكل التالى : هـذا ويمكـن تنفيـذ الأمـر Inspect عن طريق الضغط على المفتاحين Ctrl+I بلوحة المفاتيح.
Stay On Top	يعمـل هـذا الأمـر عـلى الاحتفاظ بالنافذة Local Variables ظاهرة على الشاشة حتى ولو لم تكن نشطة كما هو موضح بالشكل التالى :
Dockable	هـذا الأمـر يسمح للنافذة Local Variables بأن يكون لديها القدرة على الإرساء أو الإلتصاق بأى نافذة من النوافذ الأخرى الموجودة فى بيـئة الـتطوير المـتكاملة IDE كمـا هـو موضـح بالشـكل التالى الذى يوضـح لـنا الـنافذة Local Variables وهـى ملتصـقة بـنافذة محرر الكود البرمجى Code Editor:

		TextEditor.dpr Uses

الأمر Threads بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Threads من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح نافذة المراقبة Thread Status الموضحة فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح فتح نافذة المراقبة Thread Status وهى الضغط على مجموعة المفاتيح Ctrl+Alt+T بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح نافذة المراقبة Thread Status.

نافذة المراقبة Thread Status تعرض حالة كافة الـthreads التى يتم تنفيذها حاليا فى التطبيق الذى يكون فى مود الـDebug حاليا. وفى هذا الصدد نقول إن عملية الـ debugging للعديد من العمليات التى تتم فى نفس الوقت فيمكن تدعيمها أيضا وذلك باستخدام نافذة المراقبة Thread Status.

الأمر Modules بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Modules من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح نافذة المراقبة Modules الموضحة فى الشكل التالى :

شكل توضيحى :



هناك طريقة أخرى لفتح نافذة المراقبة Modules وهى الضغط على مجموعة المفاتيح Ctrl+Alt+M بلوحة المفاتيح.



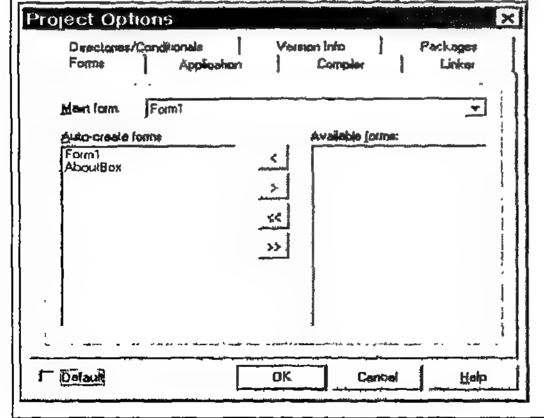
لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح نافذة المراقبة Modules.

من خلال نافذة المراقبة Modules يمكن أن نشاهد قائمة بالـModules التى تم تحميلها بالذاكرة عند تشغيل المشروع الحالى. وفى هذا الصدد نقول إن الـModule يمكن أن يكون ملف تنفيذى أو ملف مكتبة الربط الديناميكى DLL أو حزمة برمجية يحتاج المشروع الحالى أن يتم تحميلها بالذاكرة فى أثناء مرحلة التشغيل Run-Time. ونود هنا القول بأن نافذة المراقبة Modules تعرض اسم كل module وصورته فى مرحلة التشغيل Run-Time وعنوان الأساس الخاص بها بالإضافة إلى المسار الدال على موضع تحميل الـ module بالذاكرة.

من المعتاد أن يتم فتح هذه النافذة بعد أن تنتهى من ترجمة المشروع وبعد أن يتم فحصة وعلاج ما به من أخطاء وثغرات. ولكن من المفيد أيضا فتح هذه النافذة عند إجراء عملية optimizing للتقليل بقدر الإمكان من وقت التحميل وذلك عن طريق تحديد الترحيلات الأساسية للصورة المفضلة لكل Module مطلوب.



عنوان الأساس لصورة التشغيل للـModule عبارة عن offset لذاكرة الكمبيوتر ويكون بالكود السداسى عشر وهو الموضع الذى يتم فيه بشكل حقيقى تحميل الـModule كشيء مميز من عنوان الأساس للصورة المفضلة والذى يمكن تحديده من خلال صندوق الحوار Project Options الموضح فى الشكل التالى :



نافذة المراقبة Modules تتألف من الأجزاء الثلاثة التالية :

- قسم الــModule الموجودة فى الركن الأيسر العلوى بنافذة المراقبة Modules كما هو موضح بالشكل التالى :

شكل توضيحى :

Modules

Name	Base Address	Path
Process $FB578073		
TextEditor exe	$00400000	C:\Program Files\Borland\D
WINMM dll	$8FDF0000	C:\WINDOWS\SYSTEM\..
SHELL32 dll	$7FCA0000	C:\WINDOWS\SYSTEM\S
MSVCRT.dll	$78000000	C:\WINDOWS\SYSTEM\M.
SHLWAPI dll	$70BD0000	C:\WINDOWS\SYSTEM\S
comdlg32 dll	$7FE00000	C:\WINDOWS\SYSTEM\C
WINSPOOL.DRV	$7FE40000	C:\WINDOWS\SYSTEM\
COMCTL32.dll	$8FB70000	C:\WINDOWS\SYSTEM\C..
ole32 dll	$65F00000	C:\WINDOWS\SYSTEM\D.
OLEAUT32.dll	$65340000	C:\WINDOWS\SYSTEM\O
ADVAPI32 dll	$BFE80000	C:\WINDOWS\SYSTEM\A

Entry Point	Address

- قسم المصدر Source pane بالركن الأيسر السفلى بنافذة المراقبة Modules كما هو موضح بالشكل التالى :

شكل توضيحى :

Modules

Name	Base Address	Path
Process $FB57B073		
TextEditor exe	$00400000	C:\Program Files\Borland\D
WINMM dll	$8FDF0000	C:\WINDOWS\SYSTEM\
SHELL32 dll	$7FCA0000	C:\WINDOWS\SYSTEM\S
MSVCRT dll	$78000000	C:\WINDOWS\SYSTEM\M
SHLWAPI dll	$70BD0000	C:\WINDOWS\SYSTEM\S
comdlg32 dll	$7FE00000	C:\WINDOWS\SYSTEM\C
WINSPOOL DRV	$7FE40000	C:\WINDOWS\SYSTEM\
COMCTL32 dll	$8FB70000	C:\WINDOWS\SYSTEM\C
ole32 dll	$65F00000	C:\WINDOWS\SYSTEM\O
OLEAUT32 dll	$65340000	C:\WINDOWS\SYSTEM\O
ADVAPI32 dll	$8FE80000	C:\WINDOWS\SYSTEM\A

Entry Point	Address

- قسم نقطة الإدخال Entry point pane بالجانب الأيمن لنافذة المراقبة Modules كما هو موضح بالشكل التالى :

شكل توضيحى :

Modules

Name	Base Address	Path
Process $FB57B073		
TextEditor exe	$00400000	C:\Program Files\Borland\D
WINMM dll	$8FDF0000	C:\WINDOWS\SYSTEM\
SHELL32 dll	$7FCA0000	C:\WINDOWS\SYSTEM\S
MSVCRT dll	$78000000	C:\WINDOWS\SYSTEM\M
SHLWAPI dll	$70BD0000	C:\WINDOWS\SYSTEM\S
comdlg32 dll	$7FE00000	C:\WINDOWS\SYSTEM\C
WINSPOOL DRV	$7FE40000	C:\WINDOWS\SYSTEM\
COMCTL32 dll	$8FB70000	C:\WINDOWS\SYSTEM\C
ole32 dll	$65F00000	C:\WINDOWS\SYSTEM\O
OLEAUT32 dll	$65340000	C:\WINDOWS\SYSTEM\O
ADVAPI32 dll	$8FE80000	C:\WINDOWS\SYSTEM\A

Entry Point	Address

القسم Module يعرض اسم كل module وعنوانه بالموضع الذى يتم تحميله فيه بذاكرة الكمبيوتر. ولو أن الـmodule لديه معلومات عن تصحيح ومعالجة الأخطاء (debug) فى هذه الحالة نجد أن القسم Source pane يعرض مشهد شجرى للملفات المصدرية أو الأصلية التى تشتمل على الكود البرمجى الذى تم استخدامه لبناء الـModule كما أن القسم Entry point يعرض قائمة بكافة الرموز العامة كما هو موضح بالشكل التالى :

شكل توضيحى :

أمـا لو كان الـModule ليس لديه معلومات debug فى هذه الحالة نجد أن القسم Entry point يعرض نقط إدخال الدوال بالـModule كما هو موضح بالشكل التالى :

شكل توضيحى :

إجراء عملية الـ debugging لأكثر من عملية فى نفس الوقت

بالنسـبة لإمكانية إجراء عملية الـdebugging لأكثر من عملية فى نفس الوقت يتم عـرض كـل عملية وكافة الـModules المرتبطة بها وفى أثناء ذلك ستلاحظ أن العملية التى تجرى حاليا يشار إليها بسهم أخضر كما هو موضح بالشكل التالى :

شكل توضيحى :

Modules

Name	Base Address	Path
Process $FB578073		
TextEditor.exe	$00400000	C:\Program Files\Borland\D...
WINMM.dll	$BFDF0000	C:\WINDOWS\SYSTEM\...
SHELL32.dll	$7FCA0000	C:\WINDOWS\SYSTEM\S .
MSVCRT.dll	$78000000	C:\WINDOWS\SYSTEM\M...
SHLWAPI.dll	$70BD0000	C:\WINDOWS\SYSTEM\S...
comdlg32.dll	$7FE00000	C:\WINDOWS\SYSTEM\C..
WINSPOOL.DRV	$7FE40000	C:\WINDOWS\SYSTEM\...
COMCTL32.dll	$BFB70000	C:\WINDOWS\SYSTEM\C..
ole32.dll	$65F00000	C:\WINDOWS\SYSTEM\O .
OLEAUT32.dll	$65340000	C:\WINDOWS\SYSTEM\O...
ADVAPI32.dll	$BFE80000	C:\WINDOWS\SYSTEM\A..

الأمر Event Log بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر Event Log من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح نافذة المراقبة Event Log الموضحة فى الشكل التالى :

شكل توضيحى :

```
Event Log
Module Load  MSVCRT dll  No Debug Info  Base Address  $70000000  Process TextEditor exe ($FD5
Module Load  SHLWAPI dll  No Debug Info  Base Address  $70BD0000  Process TextEditor exe ($FB
Module Load  comdlg32 dll  No Debug Info  Base Address  $7FE00000  Process TextEditor exe ($FB
Module Load  WINSPOOL DRV  No Debug Info  Base Address  $7FE40000  Process TextEditor exe (
Module Load  COMCTL32 dll  No Debug Info  Base Address  $BFB70000  Process TextEditor exe ($F
Module Load  ole32 dll  No Debug Info  Base Address  $65F00000  Process TextEditor exe ($FB57B0
Module Load  OLEAUT32 dll  No Debug Info  Base Address  $65340000  Process TextEditor exe ($FE
Module Load  ADVAPI32 dll  No Debug Info  Base Address  $BFE80000  Process TextEditor exe ($FB
Module Load  GDI32 dll  No Debug Info  Base Address  $BFF20000  Process TextEditor exe ($FB576
Module Load  USER32 dll  No Debug Info  Base Address  $BFF50000  Process TextEditor exe ($FB5
Module Load  KERNEL32 dll  No Debug Info  Base Address  $BFF70000  Process TextEditor exe ($FE
Module Load  VERSION dll  No Debug Info  Base Address  $BFE70000  Process TextEditor exe ($FB
Module Load  IMM32 dll  No Debug Info  Base Address  $BFE20000  Process TextEditor exe ($FB57B
Module Load  ShellHook dll  No Debug Info  Base Address  $52000000  Process TextEditor exe ($FD
Module Load  MSVCIRT dll  No Debug Info  Base Address  $730A0000  Process TextEditor exe ($FB
Module Load  MSVCR140 dll  No Debug Info  Base Address  $77900000  Process TextEditor exe ($F
Module Load  RICHED32 dll  No Debug Info  Base Address  $7F9C0000  Process TextEditor exe ($FE
Module Load  RPCRT4 dll  No Debug Info  Base Address  $70100000  Process TextEditor exe ($FD5
First chance exception at $00417065  Exception class EFCreateError with message 'Cannot create file
```



هناك طريقة أخرى لفتح فتح نافذة المراقبة Event Log وهى الضغط على مجموعة المفاتيح Ctrl+Alt+V بلوحة المفاتيح.



لا تحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح نافذة المراقبة Event Log.

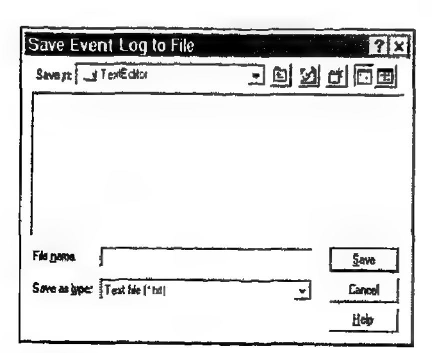
نافذة المراقبة Event Log تقدم لنا كل من الرسائل الخاصة بعملية التحكم والرسائل الخاصة بنقط التوقف ورسائل الويندوز. ونود هنا القول بأن النقر بالزر الأيمن للماوس على أى عنصر من العناصر الموجودة فى نافذة المراقبة Event Log يؤدى إلى عرض القائمة المختصرة الموضحة فى الشكل التالى :

شكل توضيحى :

```
Event Log
Module Load  COMCTL32 dll  No Debug Info  Base Address  $BFB70000  Proces
Module Load  ole32 dll  No Debug Info  Base Address  $65F00000  Process Text
Module Load  OLEAUT32 dll  No Debug Info  Base Address  $65340000  Proces
Module Load  ADVAPI32 dll  No Debug Info  Base Address  $BFE80000  Process
Module Load  GDI32 dll  No Debug Info  Base Address  $BFF20000  Process Tex
Module Load  USER32 dll  No Debug Info  Base Address  $BFF50000  Process T
Module Load  KERNEL32 dll  No Debug Info  Base Address  $BFF70000  Proces
Module Load  VERSIO      Clear Events                  $BFE70000  Proces
Module Load  IMM32       Save Events to File.          FE20000  Process Te
Module Load  ShellHo     Add Comment.                  $52000000  Process
Module Load  MSVCIF                                    $730A0000  Process
Module Load  MSVCR       Properties                    $77900000  Proce
Module Load  RICHEI                                    $7F9C0000  Proces
Module Load  RPCRT     ✓ Dockable                      70100000  Process  T
First chance exception                                 ateError with messa
```

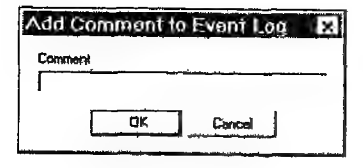
هـذه القائمـة المختصـرة تشتمل على مجموعة من الأوامر التى تسمح لك بأن تنظف نـافذة المراقـبة Event Log من محتوياتها وذلك من خلال الأمر Clear Events كما تسمح لك بأن تحفظ محتويات هذه النافذة فى ملف نصى وذلك من خلال الأمر Save Events to File والـذى يـؤدى إلى ظهـور صـندوق الحـوار Save Events to File الموضح فـى الشكل التالـى :

شكل توضيحى :



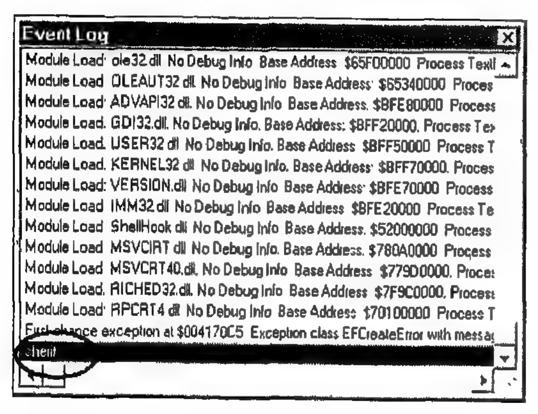
أما الأمر Add Comment فيعمل على إضافة تعليق إلى نافذة المراقبة Event Log وذلك من خلال صندوق الحوار Add Comment to Event Log الموضح فى الشكل التالــى :

شكل توضيحى :



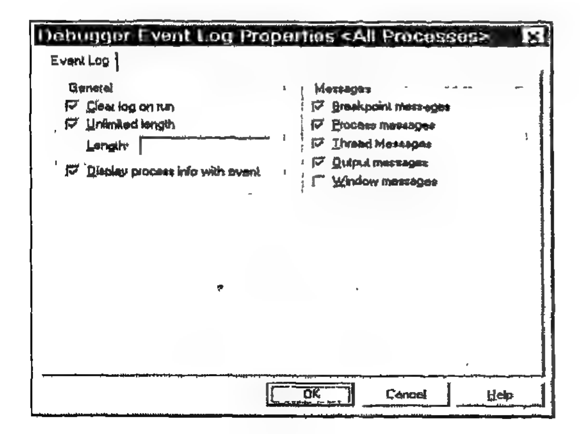
وستلاحظ أن التعليق الذى أضفته يظهر فى آخر النافذة كما هو موضح بالشكل التالى :

شكل توضيحى :



أمـا الأمـر Properties بالقائمـة المختصـرة السـالفة الذكـر فـيعمل على تحديد قيم الخصـائص الخاصـة بنافذة المراقبة Event Log وذلك من خلال صندوق الحوار الموضح فى الشكل التالى :

شكل توضيحى :



عـن طريق تحديد الخصائص تستطيع أن تتحكم فى عدد الرسائل التى يتم عرضها فـى نافذة المراقبة Event Log كما تستطيع أيضا التحكم فى نوعية الأحداث التى يتم إظهارها بهذه النافذة.

الأمر CPU بالقائمة الفرعية Debug Windows من القائمة View

اختـيار الأمـر CPU مـن القائمـة الفرعـية Debug Windows مـن القائمة View يؤدى إلى فتح النافذة CPU الموضحة فى الشكل التالى :

شكل توضيحى :



هناك طريقة أخرى لفتح النافذة CPU وهى الضغط على مجموعة المفاتيح Ctrl+Alt+C بلوحة المفاتيح.



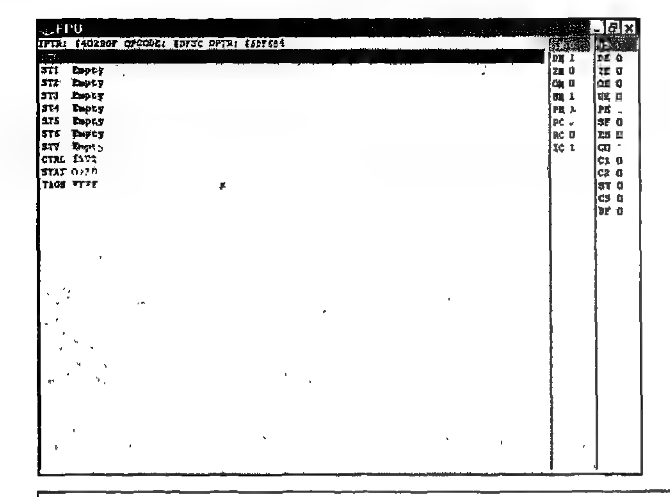
ستحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح النافذة CPU.

من خلال النافذة CPU يتم إجراء العملية debugging لمستوى معين فى التطبيق مثل محتويات حزمة البرنامج أو الـregisters أو الإشارات CPU flags أو العناصر التى ليس لها لزوم فى الذاكرة أو تعليمات التجميع المبعثرة من ماكينة الكود البرمجى الخاص بالتطبيق.

الأمر FPU بالقائمة الفرعية Debug Windows من القائمة View

اختيار الأمر FPU من القائمة الفرعية Debug Windows من القائمة View يؤدى إلى فتح النافذة CPU الموضحة فى الشكل التالى :

شكل توضيحى :





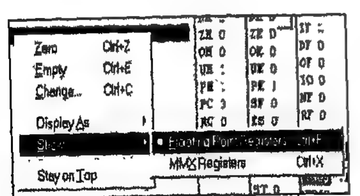
هناك طريقة أخرى لفتح النافذة FPU وهى الضغط على مجموعة المفاتيح Ctrl+Alt+F بلوحة المفاتيح.



ستحتاج لأن يكون البرنامج فى المود debug عندما ترغب فى فتح النافذة FPU.

تستطيع أن تستخدم النافذة FPU لمشاهدة محتويات المكون FPU الخاص بوحدة المعالجة المركزية CPU. هذا وتستطيع أن تجعل النافذة FPU تعرض معلومات من النوع floating-point وذلك عن طريق الضغط على المفتاحين Ctrl+F بلوحة المفاتيح أو عن طريق النقر بالزر الأيمن للماوس على أى عنصر بالنافذة ثم من القائمة المختصرة التى تظهر على الشاشة اختر الأمر Show ثم الأمر Floating-Point registers كما هو موضح بالشكل التالى :

شكل توضيحى :



كما تستطيع أن تجعل النافذة FPU تعرض معلومات من النوع MMX Registers وذلك عن طريق الضغط على المفتاحين Ctrl+X بلوحة المفاتيح أو عن طريق النقر بالزر الأيمن للماوس على أى عنصر بالنافذة ثم من القائمة المختصرة التى تظهر على الشاشة اختر الأمر Show ثم الأمر MMX Registers كما هو موضح بالشكل التالى :

شكل توضيحى :

وفى هذه الحالة تصبح المعلومات المعروضة فى النافذة FPU كما هو موضح بالشكل التالى

شكل توضيحى :

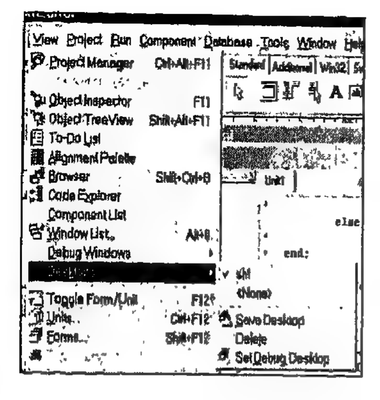
تقوم النافذة FPU بعرض كل من القيم والحالات الخاصة بكل Register فى FPU كما تعرض أيضا حالة الـFPU وأدوات التحكم وكلمات الـtags. وفى هذا الصدد نقول إن الـflags يتم تكويدها فى أداة التحكم كما يتم عرض الكلمة Status فى أجزاء أخرى منفصلة فى النافذة. كذلك تستطيع أيضا مشاهدة كل من العنوان والـopcode والـ operand التى تناظر لآخر تعليمة FPU تم تنفيذها.

الأمر Desktops بالقائمة View

من خلال الأمر Desktops الموجودة بالقائمة View نستطيع مشاهدة تخطيطات مختلفة لسطح المكتب الخاصة ببيئة التطوير المتكاملة IDE التى قمت بحفظها قبل ذلك كما تستطيع أيضا من خلال هذا الأمر حفظ تخطيطات جديدة لسطح المكتب.

الشكل التالى يوضح لنا الأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Desktops بالقائمة View :

شكل توضيحى :



الجزء العلوى بهذه القائمة الفرعية (فوق الأمر Save Desktop) يعرض التخطيطات التى سبق حفظها.

كذلك تستطيع استخدام شريط الأدوات Desktops الموضح فى الشكل التالى :

شكل توضيحى :



فمن خلال شريط الأدوات Desktops تستطيع أن تختار تخطيط سطح المكتب الذى ترغبه. فشريط الأدوات Desktops يشتمل على قائمة منسدلة تضم مجموعة من التخطيطات المختلفة لسطح المكتب كما تضم أيضا مجموعة من الأيقونات التى تسمح لك بأن تحفظ التخطيط الحالى لسطح المكتب أو لجعل التخطيط الحالى لسطح المكتب عبارة عن سطح المكتب للعملية debugging .

الأمر Save Desktop بالقائمة الفرعية للأمر Desktops بالقائمة View

لكى تحفظ التخطيط الحالى لسطح المكتب اتبع الخطوات التالية :

(١) قم بترتيب عناصر سطح المكتب بالطريقة التى ترغبها وفى أثناء ذلك يمكن أن تنقل أى عنصر وتقوم بتغيير حجم أى عنصر وتجعل عناصر ملتصقة ببعضها البعض كما هو موضح بالشكل التالى (كمثال):

شكل توضيحى :

(٢) اختر الأمر Save Desktop من القائمة الفرعية للأمر Desktops بالقائمة View أو انقر بالماوس على الأيقونة Save Current Desktop بشريط الأدوات Desktops ليظهر على الشاشة صندوق الحوار Save Desktop الموضح فى الشكل التالى :

شكل توضيحى :

(٣) فـى حقـل القائمـة المنسـدلة Save current desktop as اكتـب الاسـم الـذى ترغـب فـى تخصيصه لهذا التخطيط لعناصر سطح المكتب وليكن My Desktop كما هو موضح بالشكل التالى ثم انقر بالماوس على المفتاح Ok :

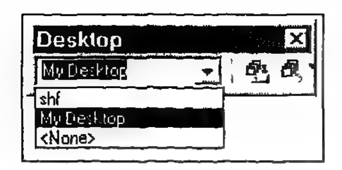
شكل توضيحى :

الآن عـندما تفـتح القائمـة View وتخـتار مـنها الأمر Desktops ستشاهد العنصر My Desktop فى الجزء العلوى بالقائمة الفرعية الخاصة بهذا الامر كما هو موضح بالشكل التالى :

شكل توضيحى :

كمـا أن القائمـة المنسـدلة الموجودة فى شريط الأدوات Desktops اصبحت مشتملة على العنصر My Desktop كما هو موضح بالشكل التالى :

شكل توضيحى :



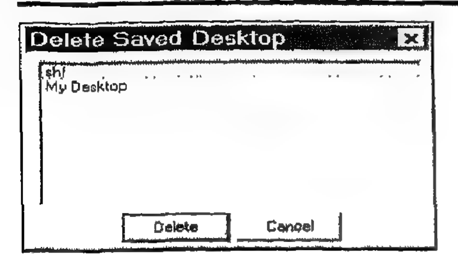
هـذا التخطـيط يـتم حفظـه لاسـتخدامه بعـد ذلـك مـع كافـة المشـاريع الـتى سيتم استخدامها عندما تبدأ فى تشغيل لغة Delphi مرة أخرى. وبهذه الطريقة تستطيع أن تنشأ ما يحلو لك من تخطيطات وتسخدمها بعد ذلك مع المشاريع التى تتولى إعدادها.

الأمر Delete بالقائمة الفرعية للأمر Desktops بالقائمة View

يعمـل الأمر Delete بالقائمة الفرعية للأمر Desktops بالقائمة View على إظهار النافذة Delete Saved Desktop الموضحة فى الشكل التالى :

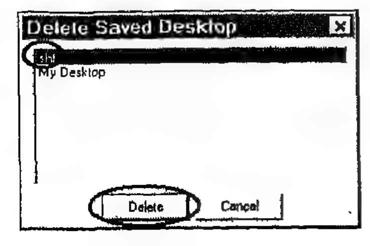


شكل توضيحى :



وهـذه الـنافذة تعـرض اسمـاء التخطـيطات الـتى سـبق حفظها ومن ثم تستطيع أن تمسـح أى مـنها وذلـك بـأن تعـلم عـلى اسـم التخطـيط الذى ترغب فى مسحه وليكن مثلا التخطـيط المسمى shf (الموضح فى الشكل التالى) ثم تنقر الماوس على المفتاح Delete كما هو موضح بالشكل التالى :

شكل توضيحى :



والآن عندما تفتح القائمة الفرعية للأمر Desktops فلن تجد بها التخطيط shf كما هو موضح بالشكل التالى :

شكل توضيحى :

كمـا لـن تجـده فـى القائمـة المنسـدلة بشـريط الأدوات Desktops كمـا هو موضح بالشكل التالى :

شكل توضيحى :



الأمر Set Debug Desktop بالقائمة الفرعية للأمر Desktops بالقائمة View

الأمر Set Debug Desktop بالقائمة الفرعية للأمر Desktops بالقائمة View يسمح لك بأن تختار واحد من تخطيطات سطح المكتب التى قمت بحفظها على أساس أن هذا التخطيط سيتم استخدامه فى كل من مرحلة التشغيل Run-Time ومرحلة الـ debugging.

عند اختيار الأمر Set Debug Desktop من القائمة الفرعية للأمر Desktops بالقائمة View يظهر على الشاشة صندوق الحوار Select debug desktop الموضح فى الشكل التالى :

شكل توضيحى :

Select debug desktop
Debug desktop
None
Cancel

صندوق الحوار Select debug desktop يعرض التخطيطات التى تستطيع أن تختار منها وذلك من خلال القائمة المنسدلة Debug desktop كما هو موضح بالشكل التالى :

شكل توضيحى :

Select debug desktop
Debug desktop:
<None>
My Desktop
<None>

لكى تحدد سطح المكتب الـ debug اختر سطح المكتب الذى ترغب فى استخدامه من أجل مرحلة الـDebugging ثم انقر بالماوس على المفتاح Ok. ومن ثم فإن سطح المكتب الـdebug يتم عرضه تلقائيا فى أثناء كافة مراحل التى تمر بها عملية الـdebugging.



عندما يخرج البرنامج من مود الـdebug ستجد أن سطح المكتب الحالى ينقلب ليصبح آخر سطح المكتب كنت تستخدمه قبل أن يدخل البرنامج فى مود الـdebug.

الأمر Toggle Form/Unit بالقائمة View

من خـلال الأمر Toggle Form/Unit بالقائمة View يمكن التنقل (ذهابا وإيابا) بـين الفورمـة الحالـية وملـف الوحـدة الخـاص بها والذى يتم عرضه فى نافذة محرر الكود البرمجى Code Editor.

هناك طرق أخرى لأداء نفس المهمة التى يقوم بها هذا الأمر وهذه الطرق عبارة عن الآتى :

- الضغط على المفتاح F12 بلوحة المفاتيح.
- النقر بالماوس على الأيقونة Toggle Form/Unit بشريط الأدوات View الموضح فى الشكل التالى :

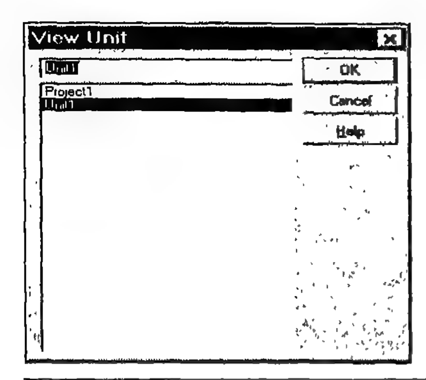
شكل توضيحى :

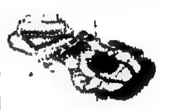


الأمر Units بالقائمة View

اختـيار الأمـر Units من القائمة View يؤدى إلى فتح صندوق الحوار View Unit الموضح فى الشكل التالى :

شكل توضيحى :





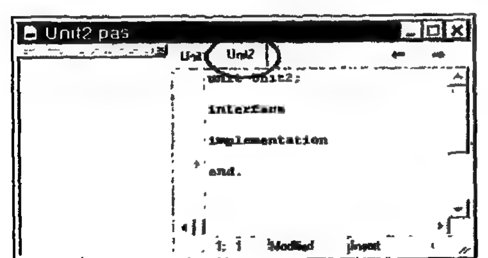
هناك طريقة أخرى لفتح صندوق الحوار View Unit وهى الضغط على المفتاحين Ctrl+F12 بلوحة المفاتيح. كما يمكن أيضا النقر بالماوس على الأيقونة View Unit بشريط الأدوات View.



صندوق الحوار View Unit

يمكن استخدام صندوق الحوار View Unit لمشاهدة ملف المشروع أو ملف أى وحدة فى المشروع الحالى. هذا وعندما تفتح وحدة فإنه تصبح الصفحة النشطة فى محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

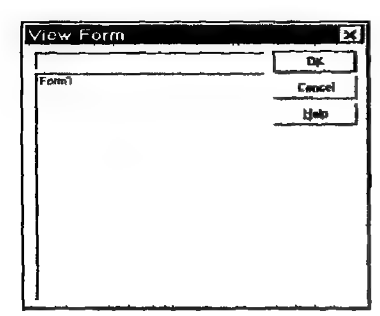


فى صندوق الحوار View Unit انقر بالماوس نقرا مزدوجا على اسم الوحدة التى ترغب فى عرضها على الشاشة. وإذا لم تكن الوحدة التى اخترتها هى الوحدة المفتوحة حاليا فستقوم اللغة على الفور بجعلها هى الوحدة المفتوحة حاليا.

الأمر Forms بالقائمة View

اختيار الأمر Forms من القائمة View يؤدى إلى فتح صندوق الحوار View form الموضح فى الشكل التالى :

شكل توضيحى :





هناك طريقة أخرى لفتح صندوق الحوار View form وهى الضغط على المفتاحين Shift+F12 بلوحة المفاتيح. كما يمكن أيضا النقر بالماوس على الأيقونة View Form بشريط الأدوات View.

صندوق الحوار View form

يمكـن استخدام صندوق الحوار View form لمشاهدة أى فورمة فى المشروع الحالى. وعـندما تخـتار فورمـة فإنها تصبح الفورمة النشطة فى المشروع كما أن ملف الوحدة الخاص بها يصبح الـModule النشط فى نافذة محرر الكود البرمجى Code Editor.

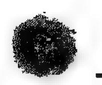
فـى صـندوق الحـوار View form انقر بالماوس نقرا مزدوجا على الفورمة التى ترغب فـى عرضـها عـلى الشاشـة وفى هذه الحالة إذا كان ملف الوحدة الخاص بهذه الفورمة غير مفـتوح عـندما اخـترت الفورمـة فسـتقوم اللغـة عـلى الفـور بفتح هذا الملف فى محرر الكود البرمجى Code Editor.

الأمر Type Library بالقائمة View

اسـتخدام الأمـر Type Library مـن القائمـة View يـؤدى إلى فـتح محـرر مكتبة الأنواع Type Library الموضح فى الشكل التالى :

شكل توضيحى :

محـرر مكتبة الأنواع Type Library تسمح لك بأن تختبر وتنشأ type information لكـل مـن أدوات الـتحكم ActiveX والخـوادم Automation والكائـنات MTS والمكونات COM الأخـرى. وأنت تستطيع توفير type information مع كائن إما مستقل stand-alone فـى أى مكتـبة مـن مكتـبات الأنـواع (المخزنة فى ملف له امتداد TLB.) كما يمكنك أيضا تكامل type information داخل ملف تنفيذى EXE. أو مكتبة من مكتبات أدوات التحكم ActiveX على أساس أنه مصدر.



عن طريق ضم مكتبة النوع مع تطبيق أو مكتبة ActiveX فإنك تصنع معلومات مشتملة فى المكتبة مثل واجهات الاستخدام Interface للكائن الخاص بها والخصائص والأساليب والأحداث والتى تصبح متاحة من تطبيق لآخر ولكافة أدوات البرمجة.

عندما تستخدم المعالجات Wizards لإنشاء أداة تحكم ActiveX أو كائن تفعيل ذاتى Automation فإنه يتم تلقائيا إنشاء مكتبة نوع. وأنت تستطيع بعد ذلك استخدام محرر مكتبة النوع Type Library لاختبار أو تعديل type information التى تم إنشاؤها بواسطة المعالج. هذا ويمكن استخدام محرر مكتبة النوع Type Library لإضافة الصفات الوظيفية مثل خصائص جديدة أو أساليب أو أحداث لمكتبة النوع التى تعدها.

الأمر New Edit Window بالقائمة View

اختيار الأمر New Edit Window من القائمة View يؤدى إلى فتح محرر جديد للكود البرمجى كما هو موضح فى الشكل التالى :

شكل توضيحى :

```
ActiveFormImpl1.pas:2
TActiveFormX
Uses
ActiveFormImpl1
unit ActiveFormImpl1;

{$WARN SYMBOL_PLATFORM OFF}

interface

uses
  Windows, Messages, SysUtils, C.
  ActiveX, AxCtrls, ActiveFormPr(

type
  TActiveFormX = class(TActiveFo
1: 1   Modified   Insert
```

ونافذة محرر الكود البرمجى الجديدة تكون مشتملة على نسخة من الصفحة النشطة من محرر الكود البرمجى Code Editor الأصلى. هذا وأى تغييرات تجريها إما فى المحرر الأصلى أو فى المحرر الجديد ينعكس على محتويات كليهما. ومن ثم تستطيع التمييز بين النوافذ حيث أن يكون الاسم الظاهرى Caption للنافذة الأصلية مشتملا على الرقم (١) ويكون الاسم الظاهرى Caption لأول نسخة مشتملا على الرقم (٢) ويكون الاسم الظاهرى Caption لثانى نسخة مشتملا على الرقم (٣) وهكذا...

الأمر Toolbars بالقائمة View

يعمل الأمر Toolbars بالقائمة View على إظهار أو إخفاء شرائط الأدوات وذلك من خلال القائمة الفرعية الخاصة به والموضحة بالشكل التالى :

شكل توضيحى :

الجدول التالى يقدم لنا وصف مختصر لكل شريط من شرائط الأدوات التالية فى بيئة التطوير المتكاملة IDE :

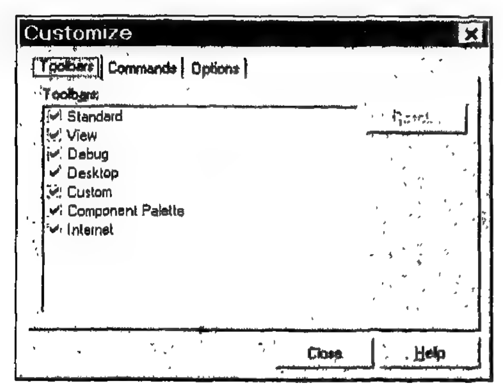
شريط الأدوات	الوصف والاستخدام
Standard	من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن إضافة عنصر جديد للمشروع الحالة وفتح أى عنصر من العناصر التاى تم حفظها قبل ذلك وحفظ المشروع الحالى وحفظ كافة المشاريع المفتوحة حاليا وفتح مشروع من المشاريع التى سبق العمل بها وكذلك إضافة ملف إلى المشروع الحالى.
View	من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن مشاهدة ملف الوحدة أو مشاهدة فورمة أو التنقل بين الفورمة وملف الوحدة أو إضافة فورمة جديدة للمشروع.
Debug	من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن تشغيل المشروع أو تعطيل تشغيله بصفة مؤقتة أو تتبع الأخطاء والثغرات بالكود البرمجي.
Custom	بشريط الأدوات هذا يمكن إضافة أى أوامر ترغبها. وفى الأصل يكون شريط الأدوات هذا مشتملا على أيقونة Help فقط.

Component Palette	شريط الأدوات هذا يتألف من عدة تبويبات تشتمل على أيقونات لكافة المكونات سواء المرئية أو الغير مرئية والتى تستخدمها لتصميم التطبيق الذى تعده.
Desktops	من خلال الأيقونات الموجودة فى شريط الأدوات هذا يمكن اختيار واحد من تخطيطات سطح المكتب المتاحة بالإضافة لحفظ التخطيط الحالى لسطح المكتب مع إمكانية جعله debug Desktop.
Internet	شريط الأدوات هذا يشتمل على أيقونات للمكونات التى تستطيع استخدامها لإنشاء تطبيقات لديها القدرة على العمل بشبكة الويب.

بالقائمة الفرعية الخاصة بالأمر Toolbars قم بالتعليم على أسماء شرائط الأدوات التى ترغب فى إظهارها على الشاشة أما شرائط الأدوات التى ترغب فى إخفاؤها من على الشاشة فقم بإلغاء التعليم من عليها بهذه القائمة الفرعية.

تستطيع أيضا تفصيل كافة شرائط الأدوات لتصبح كما ترغب أنت وهذا التفصيل يتم عن طريق إضافة أو إزالة أيقونات من شرائط الأدوات وذلك من خلال الأمر Customize بالقائمة الفرعية للأمر Toolbars والذى يعمل على فتح صندوق الحوار Customize الموضح فى الشكل التالى :

شكل توضيحى :



شريط الأدوات القياسى Standard Toolbar

الشكل التالى يوضح لنا شريط الأدوات القياسى Standard Toolbar :

شكل توضيحى :



من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات القياسى Standard Toolbar :

الأيقونة	الوظيفة
	فتح صندوق الحوار New Items الموضح فى الشكل التالى :
	فتح صندوق الحوار File Open الموضح فى الشكل التالى : وعند النقر بالماوس على السهم الموجود بهذه الأيقونة يتم فتح قائمة تضم مجموعة من المشاريع التى سبق التعامل معه كما هو موضح بالشكل التالى :



	Standard 0 C:\Program Files\Borland\Delphi6\Projects\TextEditor\TextEditor.dpr 1 C:\Program Files\Borland\Delphi6\Projects\CLXMDIApp.dpr 2 C:\Program Files\Borland\Delphi6\Projects\LOGOAPP.dpr 3 C:\Program Files\Borland\Delphi6\Projects\SDIAPP.dpr 4 C:\Program Files\Borland\Delphi6\Projects\MDIAPP.dpr 5 C:\Program Files\Borland\Delphi6\Projects\TextEditor\Unit1.pas 6 C:\Program Files\Borland\Delphi6\Projects\CLXMain.pas 7 C:\Program Files\Borland\Delphi6\Projects\LogoMain.pas 8 C:\Program Files\Borland\Delphi6\Projects\SDIMAIN.PAS 9 C:\Program Files\Borland\Delphi6\Projects\MAIN.PAS A C:\Program Files\Borland\Delphi6\Projects\W2kMain.pas B C:\PROGRAM FILES\BORLAND\DELPHI6\OBJREPOS\OKCANCL1.pas C C:\PROGRAM FILES\BORLAND\DELPHI6\OBJREPOS\OKCANCL2.pas D C:\Program Files\Borland\Delphi6\Projects\Unit3.pas E C:\Program Files\Borland\Delphi6\Projects\Unit4.pas
	يسمح لك بأن تخزن التغييرات التي أجريتها على كافة الملفات الموجودة بالمشروع الذى يتم التعامل معه حاليا باستخدام الاسم الحالى لكل ملف (كما يمكن أيضا الضغط على المفتاحين Ctrl+S بلوحة المفاتيح).
	يسمح لك بحفظ كافة الملفات المفتوحة حاليا بما فيها ملفات المشروع الحالى والـ modules (كما يمكن أيضا الضغط على المفاتيح Shift+Ctrl+S بلوحة المفاتيح).
	يسمح لك بأن تفتح مشروع من المشروعات التي سبق التعامل معها (كما يمكن أيضا الضغط على المفتاحين Ctrl+F11 بلوحة المفاتيح).
	يفتح صندوق الحوار Add to Project الموضح فى الشكل التالى : Add to Project Look in: Projects Bpl, TextEditor, About, ActiveFormImpl1, ActiveFormProj1_TLB, Childwin, CLXAbout, CLXChildWin, CLXMain, LogoMain, Main, Sdimain, Unit1, Unit2, Unit3, Unit4, Unit5, W2kMain File name: ...gram Files\Borland\Delphi6\Projects\W2kMain — Open Files of type: Delphi unit (*.pas) — Cancel والذى من خلاله يمكن إضافة ملف إلى المشروع الذى يجرى التعامل معه حاليا(كما يمكن أيضا الضغط على المفتاحين Shift+F11 بلوحة المفاتيح).
	يفتح صندوق الحوار Remove from Project الموضح فى الشكل التالى :

	Remove From Project Unit Form About AboutBox W2kMain Win2kAppForm OK Cancel Help والذى يمكن من خلاله مسح ملف من المشروع الذى يجرى التعامل معه حاليا.

شريط الأدوات View Toolbar

الشكل التالى يوضح لنا شريط الأدوات View Toolbar:

شكل توضيحى :



مـن خـلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات View Toolbar:

الأيقونة	الوظيفة
	يفتح صندوق الحوار View Unit الموضح فى الشكل التالى : View Unit W2kMain About W2kMain Win2kApp OK Cancel Help

	والذى يكن من خلاله مشاهد محتويات وحدة من الوحدات الموجودة بالمشروع (كما يمكن أيضا الضغط على المفتاحين Ctrl+F12 بلوحة المفاتيح).
	يفتح صندوق الحوار View Form الموضح فى الشكل التالى : والـذى يمكـن من خلاله عرض فورمة من الفورم الموجودة بالمشروع (كما يمكن أيضا الضغط على المفتاحين Shift+F12 بلوحة المفاتيح).
	التنقل بين الفورمة وملف الوحدة الخاصة بها فى محرر الكود البرمجى Code Editor (كمـا يمكـن أيضـا الضـغط عـلى المفـتاح F12 بلوحـة المفاتيح).
	إضافة فورمة جديدة إلى المشروع.

شريط الأدوات Debug Toolbar

الشكل التالى يوضح لنا شريط الأدوات Debug Toolbar:

شكل توضيحى :



مـن خـلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات Debug Toolbar:

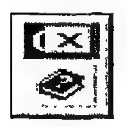
الأيقونة	الوظيفة
	تشـغيل وتـرجمة المشـروع الـذى يتم العمل به حاليا وعند النقر بالماوس عـلى السـهم الموجـود بهـذه الأيقونـة يـتم فـتح قائمة تضم مجموعة من المشـاريع المفـتوحة حالـيا لكـى تخـتار مـنها كمـا هـو موضح بالشكل التالـى : وهذا الأمر يناظر الأمر Run الموجود بالقائمة Run (يمكن أيضا الضغط على المفتاح F9 بلوحة المفاتيح).
	تعطـيل تشـغيل المشروع بصفة مؤقتة وهذا الأمر يناظر الأمر Program Pause الموجود بالقائمة Run.
	تتـبع الأخطـاء والـثغرات فى الكود البرمجى سطر بسطر كما هو موضح بالشكل التالى :

(يمكـن أيضـا الضـغط على المفتاح F7 بلوحة المفاتيح) وهذا الأمر يناظر الأمر Trace Into الموجود بالقائمة Run	
تخطـى خطـوة فـى أثـناء تتـبع الأخطـاء والثغرات فى الكود البرمجـى (يمكن أيضا الضغط على المفتاح F8 بلوحة المفاتيح).	

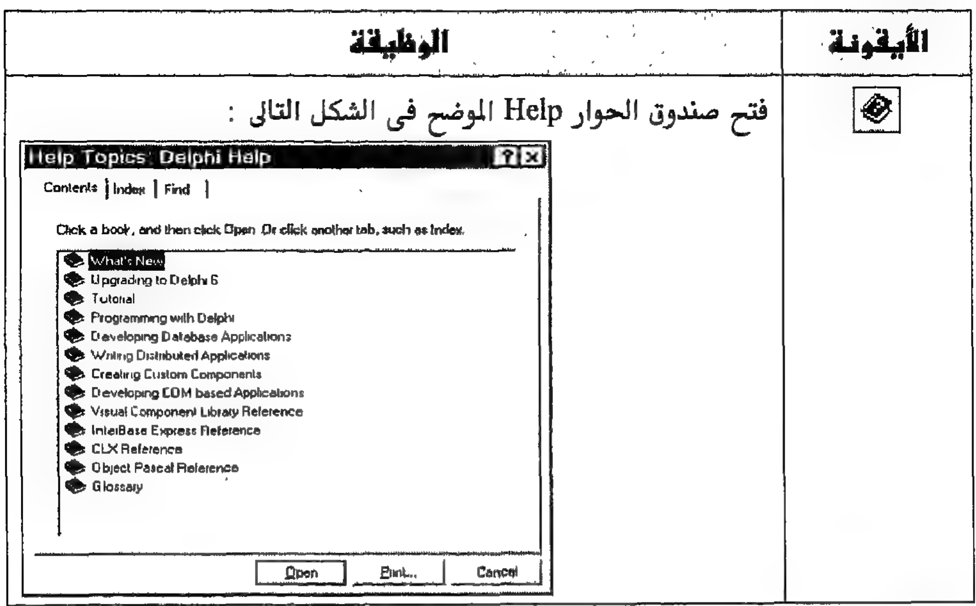
شريط الأدوات Custom Toolbar

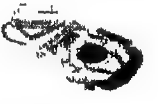
الشكل التالى يوضح لنا شريط الأدوات Custom Toolbar:


شكل توضيحى :



مـن خـلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود في شريط الأدوات Custom Toolbar:

الوظيفة	الأيقونة
فتح صندوق الحوار Help الموضح فى الشكل التالى : 	



بالرغم من أن شريط الأدوات Custom Toolbar يشتمل على الأيقونة Help فقط إلا إنه من المكن إضافة أى أيقونات لشريط الأدوات هذا.

شريط الأدوات Desktops Toolbar

الشكل التالى يوضح لنا شريط الأدوات Desktops Toolbar:

شكل توضيحى :



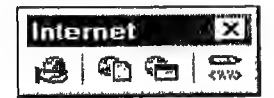
مـن خـلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات Desktops Toolbar:

الأيقونة	الوظيفة
My Desktop	تسمح لك بأن تتحول إلى أى من التخطيطات المتعددة لسطح المكتب والتى تم إنشاؤها حفظها من قبل.
	عرض صندوق الحوار Save Desktop الموضح فى الشكل التالى : Save Desktop Save current desktop as: My Desktop OK Cancel والذى يمكن من خلاله تحديد الاسم الذى ترغب فى استخدامه لحفظ تخطيط سطح المكتب الذى أنشأته.
	جعل التخطيط الحالى لسطح المكتب هو نفس تخطيط سطح المكتب عندما يكون البرنامج فى المود Debug أو عندما يكون فى مرحلة التشغيل Run-Time.

شريط الأدوات Internet Toolbar

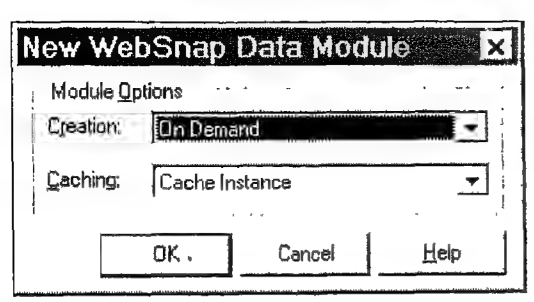
الشكل التالى يوضح لنا شريط الأدوات Internet Toolbar:

شكل توضيحى :



من خلال الجدول التالى نتعرف على وظيفة الأيقونات الموجود فى شريط الأدوات Internet Toolbar:

الأيقونة	الوظيفة
New WebSnap Application	يعرض صندوق الحوار New WebSnap Application الموضح فى الشكل التالى : New WebSnap Application Server Type ISAPI/NSAPI Dynamic Link Library CGI Stand-alone executable Win-CGI Stand-alone executable Apache Shared Module (DLL) Web App Debugger executable Application Module Components Page Module Data Module Components... Application Module Options Page Name: PageProducerPage2 Page Options... Caching: Cache Instance Default OK Cancel Help وصندوق الحوار هذا يعتبر أول خطوة فى عملية إنشاء تطبيق يعمل فى خوادم شبك الويب Web Server مع Web Snap كما يمكن استخدام نفس صندوق الحوار هذا فى عملية تحويل أى تطبيق سبق إنشاؤه ليصبح من النوع الذى يمكن أن يعمل بخوادم شبكة الويب.
New WebSnap Page Module	يعرض صندوق الحوار WebSnap Page Module الموضح فى الشكل التالى :

	New WebSnap Page Module Producer Type: PageProducer Script Engine: JScript HTM New File: Template: Standard Page Name: PageProducerPage2 Title: PageProducerPage2 Published: Login Required: Module Options Creation: On Demand Caching: Cache Instance Default OK Cancel Help يمكن استخدام صندوق الحوار هذا لإنشاء صفحة محتوى. ونود هنا القول بأن Module الصفحة يكون مشتملا على الأداة page producer التى تكون مسئولة عن تكوين محتويات أى صفحة.
New WebSnap Data Module	عرض صندوق الحوار New WebSnap Data Module الموضح فى الشكل التالى :  من خلال صندوق الحوار هذا يتم إنشاء Module بيانات فارغ بحيث يتم فيه إضافة المكونات الخاصة بـ WebSnap Internet. وفى هذا الصدد نقول إن اختيار هذا النوع من الـModule لا يؤدى إلى إنشاء صفحة محتوى. وهذا الـModule يتم استخدامه كما لو كان حاوية للمكونات المتاحة للاستخدام المشترك بواسطة الـModules الأخرى–كمكونات قواعد البيانات المستخدمه من خلال الإثنين.

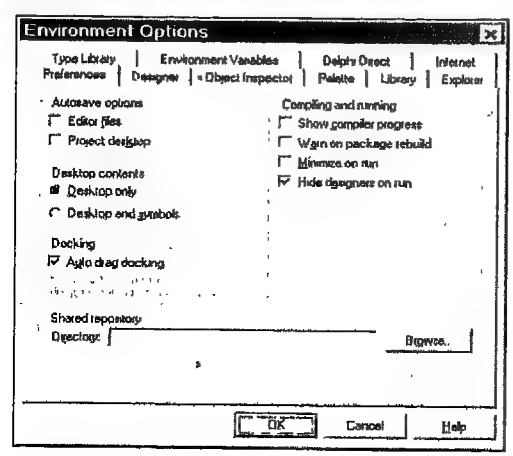
استخدام المحرر الخارجى External Editor

لكـى تـتمكن مـن الولوج إلى المحرر الخارجى External Editor ينبغى عليك أولا تهيئتة وإعداده.

خطوات إعداد وتهيئة المحرر الخارجى External Editor

(١) افـتح القائمـة Tools واخـتر منها الأمر Environment Options ليظهر على الشاشة صندوق الحوار

شكل توضيحى :



(٢) انقر بالماوس على التبويب Internet ليظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :

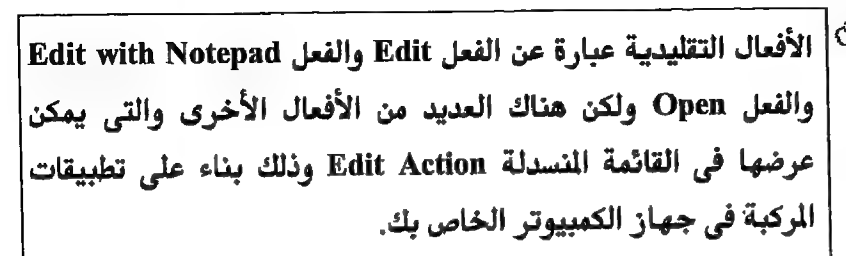
(٣) انقر بالماوس على المفتاح New ليظهر على الشاشة صندوق الحوار New Type الموضح فى الشكل التالى :

شكل توضيحى :

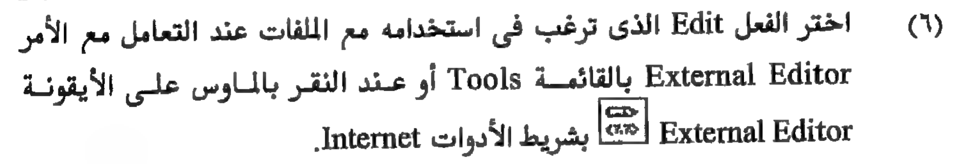
New Type
Description:
Extensions:
Edit Action:
Syntax Highlight
Show HTML Custom View
Show XML Custom View
OK Cancel Help

(٤) بالحقل Description اكتب وصف مختصر للمحرر الخارجى ثم فى الحقل Extensions اكتب امتدادات الملفات التى ترغب فى فتحها من خلال المحرر الخارجى

(٥) فى القائمة المنسدلة Edit Action التى تعرض أفعال التحرير المتاحة والمخزنة فى الـRegistry الخاص بجهازك أو امتدادات الملفات التى أدخلتها فى الحقل Extensions.



الأفعال التقليدية عبارة عن الفعل Edit والفعل Edit with Notepad والفعل Open ولكن هناك العديد من الأفعال الأخرى والتى يمكن عرضها فى القائمة المنسدلة Edit Action وذلك بناء على تطبيقات المركبة فى جهاز الكمبيوتر الخاص بك.



(٦) اختر الفعل Edit الذى ترغب فى استخدامه مع الملفات عند التعامل مع الأمر External Editor بالقائمة Tools أو عند النقر بالماوس على الأيقونة External Editor بشريط الأدوات Internet.

(٧) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار New Type.

(٨) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار Environment Options.



خطوات فتح المحرر الخارجى External Editor

بمجرد أن تفتح ملف ينتمى لأنواع الملفات التى حددتها من خلال صندوق الحوار New Type تستطيع إذا أن تفتح المحرر الخارجى External Editor من خلال الخطوات التالية :

(١) اختر الأمر External Editor من القائمة Tools أو انقر بالماوس على الأيقونة External Editor بشريط الأدوات Internet.

(٢) ستجد أن تطبيق الويب الذى تتعامل معه قد قام بفتح المحرر الذى قمت بتركيبه قبل ذلك.

القائمة Project Menu

يمكن استخدام القائمة Project لترجمة أو بناء التطبيق الذى تعده. هذا وينبغى أن يكون لديك مشروع واحد على الأقل مفتوح.

من خلال الجدول التالى سنتعرف سويا باختصار على وظيفة كل أمر من الأوامر الموجودة فى القائمة Project :

الأمر	الوصف والاستخدام
Add to Project	إضافة ملف إلى المشروع الحالى.
Remove from project	إزالة ملف من المشروع الحالى.
Import Type Library	استيراد واحدة من مكتبات الأنواع للمشروع الذى يتم التعامل معه حاليا.
Add to Repository	إضافة المشروع الذى ترغبه إلى مخزن الكائنات Object Repository.
View Source	عرض ملف المشروع فى نافذة محرر الكود البرمجى Code Editor.
Languages	يسمح لك بأن تضيف أو تلغى أو تقوم بتحديث الملفات DLL المصدرية أو تختار لغة من أجل الإختبار.

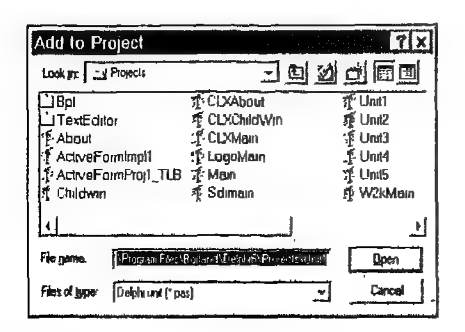
Add New Project	فتح صندوق الحوار New Items الذى يحتوى على عدد من المعالجات والكائنات المخزنة فى مخزن الكائنات. وانت تستطيع إما أن تقوم بتكوين كائن جديد أو البدء مع أى كائن من الكائنات السابقة الإعداد والمخزنة فى مخزن الكائنات.
Add Existing Project	من خلال هذا الأمر يمكن استخدام صندوق الحوار Open Project لإضافة مشروع موجود بالفعل إلى مدير المشروع.
Compile project	يقوم هذا الأمر بترجمة الملفات الموجودة فى المشروع الحالى فقط والتى حدث لها تغغيرات منذ أن تم بناؤها آخر مرة.
Build project	ترجمة أى عنصر فى المشروع بغض النظر عما إذا كان حدث تغيير فى المصدر أم لا.
Syntax Check project	ترجمة المشروع الذى تعده ولكنه لا يقوم بربطه مع مكتبات اللغة.
Information for project	عرض كافة معلومات البناء و حالته بالنسبة للمشروع الذى تتعامل معه.
Compile All Projects	ترجمة أى كود برمجى مصدرى تم تغييره منذ آخر ترجمة فى كافة المشروعات الموجودة فى مجموعة المشروع.
Build All Projects	ترجمة أى شيئ فى مجموعة المشروع بغض النظر عن عما إذا كان حدث تغيير فى الأصول أم لا.
Web Deployment Options	إعداد القيم التحديدية Settings الضرورية لنشر وتوزيع أدوات التحكم ActiveX أو الفورم Active Form التى تم الإنتهاء من إعدادها.
Web Deploy	بعد تحديد الخيارات الخاصة بالنشر عبر شبكة الويب وترجمة المشروع يقوم هذا الأمر بنشر أدوات التحكم ActiveX أو الفورم Active Form التى إنتهيت من إعدادها

Options	عـرض صـمدوق الحـوار Project Options والـذى يمكن من خلاله الـتعامل مـع الاختـيارات الخاصـة بـك مـن عملية الترجمة وعملية الـربط والفـورم الإفتراضـية ومعلومات الإصدار...بالإضافة إلى العديد من العناصر الأخرى.

الأمر Add to Project بالقائمة Project

اختـيار الأمـر Add to Project من القائمة Project يؤدى إلى فتح صندوق الحوار Add To Project الموضح فى الشكل التالى :

شكل توضيحى :





> يمكن أيضا فتح صندوق الحوار **Add To Project** عن طريق الضغط على المفتاحين **Shift+F11** بلوحة المفاتيح.

صندوق الحوار Add To Project

يمكـن اسـتخدام صـندوق الحـوار Add To Project لإضـافة وحـدة مـن الوحدات الموجـودة بـالفعل كمـا يـتم أيضا إضـافة الفورمة الملحقة بالوحدة المراد إضافتها إلى المشروع الذى تعمل به حاليا. هذا وعندما تضيف وحدة إلى مشروع تجد أن اللغة تقوم تلقائيا بإضافة هذه الوحدة إلى الجملة uses بملف المشروع.

الجـدول الـتالى يقدم لـنا وصـف مختصر للأجزاء التى يتألف منها صندوق الحوار Add To Project:

المعامل	الوصف والاستخدام
Look In	من خلال هذه القائمة المنسدلة نستطيع اختيار مشغل الأقراص أو الفهرس الذى سيتم البحث فيه عن الملفات المطلوب إضافتها للمشروع كما هو موضح بالشكل التالى : بعد ذلك تعرض نافذة الملف الفهارس الفرعية الموجودة داخل مشغل الأقراص المختار أو داخل الفهرس المختار كما إنها تعرض أيضا كافة الملفات التى تتطابق أنواعها مع الأنواع المحددة بالفلتر الموجود بالقائمة المنسدلة Files of Type.
File Window	نافذة الملف تعرض الفهارس الفرعية الموجودة بالفهرس المفتوح حاليا بالإضافة إلى الملفات الموجودة فى الفهرس المفتوح حاليا والتى تتطابق مع الفلتر المطبق حاليا بالقائمة المنسدلة Files of Type كما هو موضح بالشكل التالى :



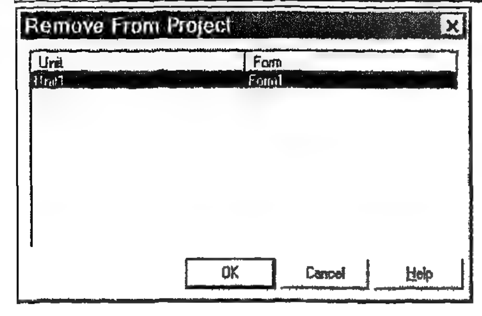
	اختر الملف الذى ترغب فى إضافته إلى المشروع وذلك عن طريق النقر عليه بالماوس أو باستخدام مفاتيح الأسهم الموجودة فى لوحة المفاتيح أو بالنقر بالزر الأيمن للماوس على الملفات التى ترغبها وذلك لعرض القائمة المختصرة الموضحه فى الشكل التالى : ومنها اختر الأمر Select.
File Name	بهذا الحقل يمكن أن تكتب اسم الملف الذى ترغب فى إضافته للمشروع كما يمكن أيضا أن تكتب فلتر ext.* للحد من الملفات التى تظهر فى نافذة الملف كما هو موضح بالشكل التالى :
Files of type	من خلال هذه القائمة المنسدلة تستطيع أن تختار الفلتر الذى يعمل على تحديد نوعية الملفات التى تظهر فى المنطقة File Window. هذا والشكل التالى يوضح لنا الفلاتر الموجودة بهذه القائمة المنسدلة :

Up on Level	النقر على هذه الأيقونة يؤدى إلى الإنتقال للمستوى الأعلى فى الهيكل الشجرى الموجود به الفهرس المفتوح حاليا.
Create New Folder	النقر على هذه الأيقون يعمل على إنشاء فهرس جديد ليكون فهرس فرعى بالفهرس أو مشغل الأقراص المفتوح حاليا كما هو موضح بالشكل التالى :
List	هذه الأيقونة تعمل على عرض أسماء الملفات فقط فى المنطقة File Window وهذا هو الوضع الطبيعى لطريقة العرض.
Details	هذه الأيقونة تعمل على عرض معلومات تفصيلية عن كل عنصر موجود فى المنطقة File Window كالأسم والحجم كما هو موضح بالشكل التالى :

الأمر Remove from Project بالقائمة Project

اختيار الأمر Remove from Project من القائمة Project يؤدى إلى فتح صندوق الحوار Remove from project الموضح فى الشكل التالى :

شكل توضيحى :



من خلال صندوق الحوار Remove from project تستطيع اختيار العنصر الذى ترغب فى إزالته من المشروع ثم النقر بالماوس على المفتاح Ok.

الأمر Import Type Library بالقائمة Project

اختيار الأمر Import Type Library من القائمة Project يؤدى إلى فتح صندوق الحوار Import Type Library الموضح فى الشكل التالى :

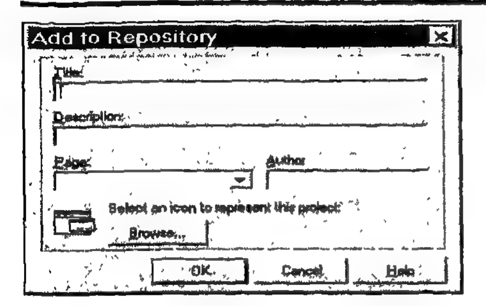
شكل توضيحى :

من خلال صندوق الحوار Import Type Library تستطيع اختيار مكتبة النوع التى ترغب فى استيرادها ثم تنقر بالماوس على المفتاح Create Unit.

الأمر Add To Repository بالقائمة Project

اختيار الأمر Add To Repository من القائمة Project يؤدى إلى فتح صندوق الحوار Add To Repository الموضح فى الشكل التالى :

شكل توضيحى :



عن طريق صندوق الحوار Add To Repository يمكن أن تضيف المشاريع والفورم التى ترغبها إلى مخزن الكائنات.

الأمر View Source بالقائمة Project

اختيار الأمر View Source من القائمة Project يؤدى إلى عرض ملف المشروع للمشروع الحالى وذلك فى نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

```
program Project1;

uses
  Forms,
  Unit1 in 'Unit1.pas' {Fo

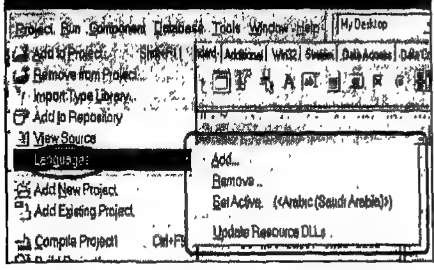
{$R *.res}

begin
  Application.Initialize;
```

الأمر Languages بالقائمة Project

القائمة الفرعية الخاصة بالأمر Languages بالقائمة Project والموضحة فى الشكل التالى تشتمل على عدد من الاختيارات التى تعمل على تغيير القيم التحديدية الخاصة بترجمة المشروع الذى تتعامل معه :

شكل توضيحى :

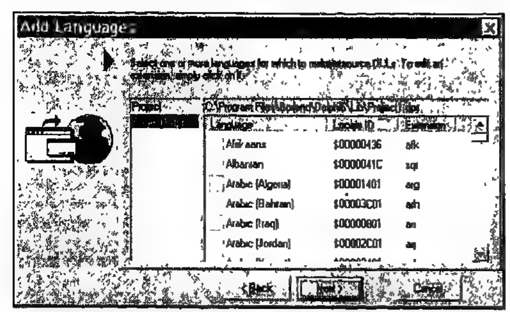


الأمر Add بالقائمة الفرعية للأمر Languages بالقائمة Project

يعمل الأمر Add بالقائمة الفرعية للأمر Languages بالقائمة Project على فتح المعالج Add Languages وفيه يظهر أول صندوق حوار كما هو موضح بالشكل التالى :

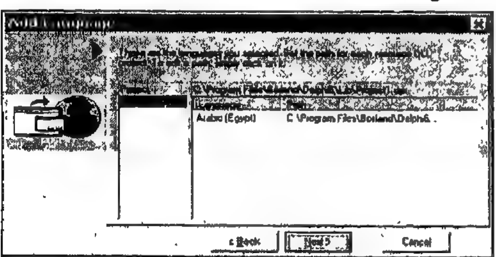
شكل توضيحى :

فى صندوق الحوار هذا تختار واحد أو أكثر من المشروعات التى تصنع مكتبات الإتصال الديناميكى DLLs وفيه انقر بالماوس على المفتاح Select All ثم على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :



شكل توضيحى :

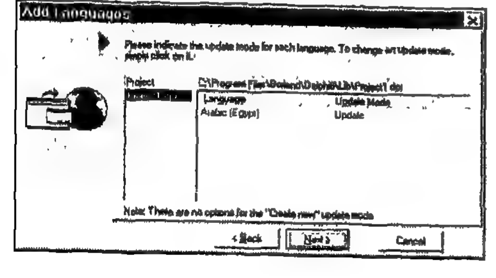
فى صندوق الحوار هذا تستطيع أن تختار لغة واحد أو أكثر لاستخدامها فى إعداد المكتبات DLL وفيه علم على المربع المجاور للغة Arabic (Egypt) ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :



شكل توضيحى :

بصندوق الحوار هذا تشاهد اللغات التى اخترتها والمطلوب منك الآن تحديد مسار لكل ملف من الملفات DLLs ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :

شكل توضيحى :



صندوق الحوار هذا يطلب منك تحديد مود التحديث لكل لغة وبعد ذلك انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :

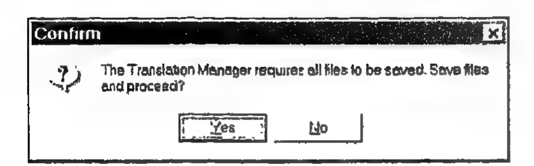
شكل توضيحى :

يطلب منك صندوق الحوار هذا أن تنقر بالماوس على المفتاح Finish للبدء فى إنشاء الملفات DLLs للمشروع الذى تعمل به. بعد ذلك يظهر على الشاشة صندوق الحوار Processed الذى يقدم لك تقرير مختصر عن عدد الملفات والعناصر التى تم التعامل معها فى أثناء الإنشاء كما هو موضح بالشكل التالى :

شكل توضيحى :

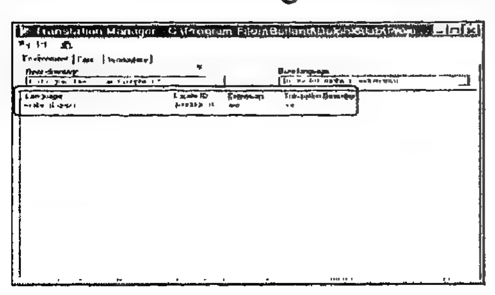
بعد أن تنتهى من قراءة التقرير انقر بالماوس على المفتاح Ok لتظهر الرسالة الموضحة فى الشكل التالى :

شكل توضيحى :



هذه الرسالة تقول لك أن مدير الترجمة يطلب منك ضرورة أن تقوم بحفظ كافة الملفات فهل أنت موافق على القيام بذلك. انقر بالماوس على المفتاح Yes. والآن تظهر على الشاشة نافذة مدير الترجمة وبها اللغة التى اخترتها كما هو موضح بالشكل التالى :

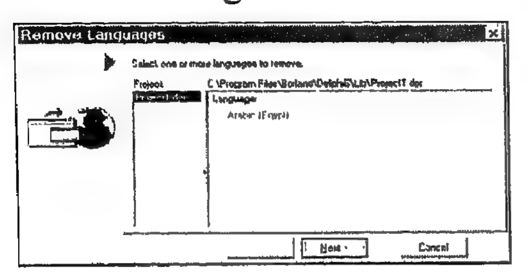
شكل توضيحى :



الأمر Remove بالقائمة الفرعية للأمر Languages بالقائمة Project

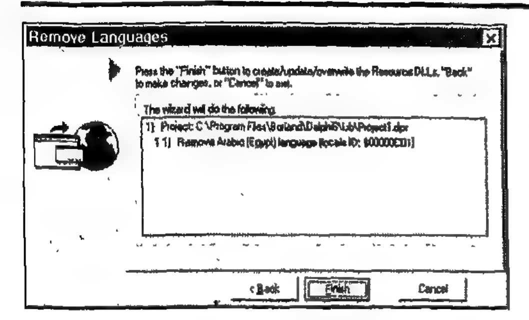
يعمل الأمر Remove بالقائمة الفرعية للأمر Languages بالقائمة Project على فتح المعالج Remove Languages وفيه يظهر أول صندوق حوار كما هو موضح بالشكل التالى :

شكل توضيحى :



صندوق الحوار هذا يطلب منك أن تختار واحدة أو أكثر من اللغات المطلوب إزالتها وفيه علم بالماوس على اللغة Arabic (Egypt) ثم انقر بالماوس على المفتاح Next للانتقال لصندوق الحوار التالى والموضح فى الشكل التالى :

شكل توضيحى :

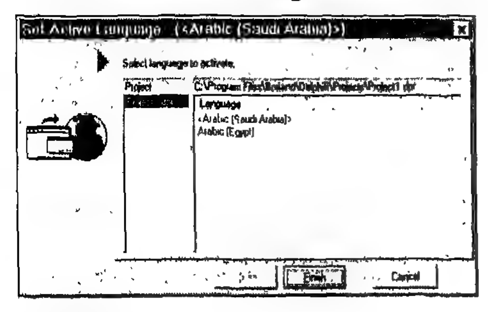


بصندوق الحوار هذا انقر بالماوس على المفتاح Finish ليتم مسح الملفات DLLs الخاصة باللغة المطلوب إزالتها وللقيام أيضا بتحديث المشروع.

الأمر Set Active بالقائمة الفرعية للأمر Languages بالقائمة Project

من خلال الأمر Set Active بالقائمة الفرعية للأمر Languages بالقائمة Project يعمل على يمكن اختيار DLL مصدرى من أجل الاختبار. هذا وعند استخدام هذا الأمر يظهر على الشاشة صندوق الحوار Set Active Language الموضح فى الشكل التالى :

شكل توضيحى :



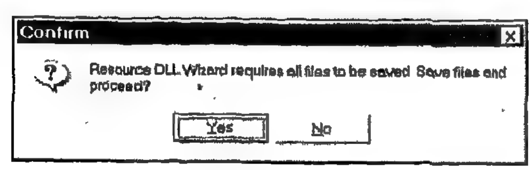
صندوق الحوار هذا يطلب منك اختيار اللغة التى ترغب فى تفعيلها وفيه اختر Arabic (Egypt) ثم انقر بالماوس على المفتاح Finish. بعد ذلك ستلاحظ حدوث بعض التغييرات فى الكود البرمجى الموجود فى محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

الأمر Update Resource DLLs بالقائمة الفرعية للأمر Languages بالقائمة Project

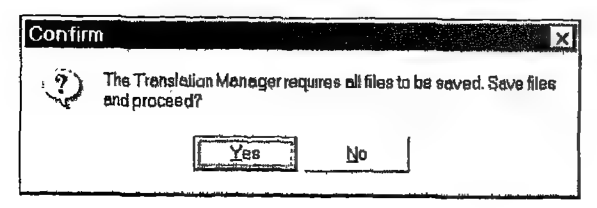
من خلال الأمر Update Resource DLLs بالقائمة الفرعية للأمر Languages بالقائمة Project يمكن تحديث الملفات DLLs التى تعتبر أحد مصادر المشروع. هذا وعند التعامل مع هذا الأمر تظهر الرسالة الموضحة فى الشكل التالى :

شكل توضيحى :



هذه الرسالة تقول لك أن المعالج Resource DLL Wizard يحتاج لأن يتم حفظ كافة الملفات المفتوحة الآن ومن ثم انقر بالماوس على المفتاح Yes. بعد ذلك تظهر الرسالة الموضحة فى الشكل التالى :

شكل توضيحى :

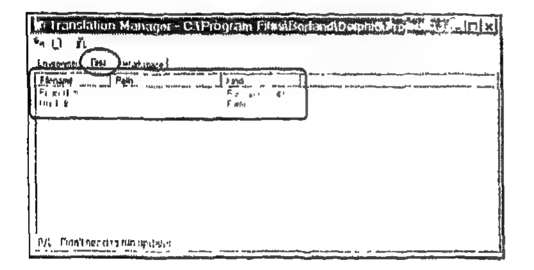


هذه الرسالة تقول لك أن مدير الترجمة Translation Manager يطلب أن يتم حفظ كافة الملفات المفتوحة كلها ومن ثم انقر بالماوس على المفتاح Yes. الآن تظهر على الشاشة نافذة مدير الترجمة Translation Manager وبها نشاهد ثلاثة تبويبات هى : تبويب بيئة العمل Environment وفيه نشاهد اللغة التى اخترناها وبعض المعلومات عنها كما هو موضح بالشكل التالى :

شكل توضيحى :

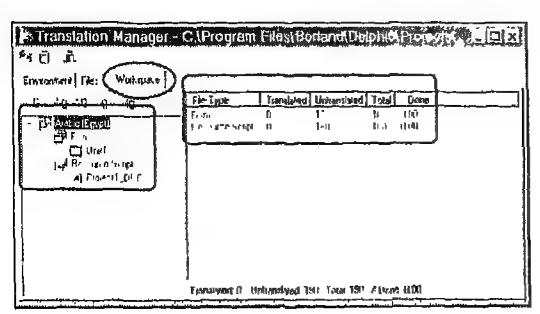
أمـا التبويـب Files فـيعرض الملفـات الخاصـة بالمشـروع ونوعـية كل منها كما هو موضح بالشكل التالى :

شكل توضيحى :



وأخيرا فتبويب منطقة العمل Workspace فيعرض كيفية ارتباط اللغة المختارة مع ملفات المشروع كما هو موضح بالشكل التالى :

شكل توضيحى :



الأمر Add New Project بالقائمة Project

يـتم اسـتخدام الأمر Add New Project من القائمة Project لإضافة عنصر جديد (مـثل تطبـيق جديـد أو حـزمة برمجية أو ملف DLL...) إلى مجموعة المشروع. ويعمل هذا الأمر على فتح صندوق الحوار New Items الموضح فى الشكل التالى :

شكل توضيحى :

من خلال صندوق الحوار هذا يمكن أن تختار نوعية المشروع الذى ترغب فى إضافته إلى مجموعة المشروع.

الأمر Add Existing Project بالقائمة Project

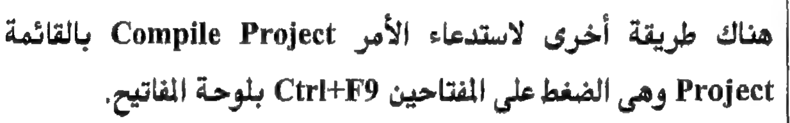
يعمل الأمر Add Existing Project بالقائمة Project على إضافة مشروع من المشروعات التى سبق إعدادها إلى مجموعة المشروع الحالى وذلك من خلال صندوق الحوار Open Project الموضح فى الشكل التالى :

شكل توضيحى :

الأمر Compile Project بالقائمة Project

يعمل الأمر Compile Project بالقائمة Project على ترجمة كافة الملفات الموجودة فى المشروع الحالى والتى حدث بها تغيير منذ آخر بناء لها وهذه الترجمة ينتج عنها عدة أنواع من الملفات مثل الملفات التنفيذية EXE وملفات الربط الديناميكى DLLs وملفات المصادر RES وغيرها.





هناك طريقة أخرى لاستدعاء الأمر Compile Project بالقائمة Project وهى الضغط على المفتاحين Ctrl+F9 بلوحة المفاتيح.

الأمر Build Project بالقائمة Project

اختيار الأمر Build Project من القائمة Project يؤدى إلى بناء (أو إعادة بناء) كافة ملفات المشروع الحالى بغض النظر عما إذا كان حدث لها تغيير أم لا منذ آخر ترجمة

وبناء لها. وهذا الأمر يكون مفيد لك خاصة عندما تقوم بتغيير الموجهات العامة الخاصة بمترجم اللغة أو إذا كنت أجريت تغييرات على خيارات المترجم وفى هذه الحالة يجعلك هذا الأمر تتأكد من إن كافة الأكواد البرمجية قد تأثرت فعلا بهذه التغييرات.

الأمر Syntax check Project بالقائمة Project

اختيار الأمر Syntax check Project من القائمة Project يؤدى إلى ترجمة الـ Modules الخاصة بالمشروع الذى تتولى إعداده ولكنه لا يقوم بربطها. ومن خلال هذا الأمر تتوفر لك الوسائل التى تمكنك من مراجعة الكود البرمجى الذى تكتبه.

الأمر Information for Project بالقائمة Project

اختيار الأمر Information for Project من القائمة Project يؤدى إلى فتح صندوق الحوار Information الموضح فى الشكل التالى :

شكل توضيحى :

Information

Program	
Source compiled:	41 lines
Code size:	311116 bytes
Data size:	6985 bytes
Initial stack size:	16384 bytes
File size:	358424 bytes

Packages Used
(None)

Status
Project1 Successfully Compiled.

OK Help

من خلال صندوق الحوار هذا نحصل على معلومات حول المشروع الذى يتم العمل به وهذه المعلومات تتمثل فى عدد سطور الكود البرمجى وحجمه بالبايت وحجم البيانات بالبايت وحجم الملف وحالة ترجمة المشروع.



الأمر Information for Project لا يكون متاح للاستخدام إلا بعد أن يتم بناء أو ترجمة المشروع من خلال الأمر Build Project أو الأمر Compile Project.

الأمر Compile All Projects بالقائمة Project

يعمل الأمر Compile All Projects بالقائمة Project على ترجمة الملفات الموجودة بكافة المشاريع الموجودة بمجموعة المشروع علما بأنه يتم ترجمة الملفات التى حدث لها تغيير منذ آخر عملية بناء. ونود هنا القول بأن الأمر Compile All Projects يشبه إلى حد كبير الأمر Build All Projects والفرق الوحيد بينهما يتمثل فى أن الأمر Build All Projects يقوم بترجمة كافة الملفات بغض النظر عما إذا كان حدث لها تغيير أم لا.

الأمر Build All Projects بالقائمة Project

يعمل الأمر Build All Projects بالقائمة Project على ترجمة الملفات الموجودة بكافة المشاريع الموجودة بمجموعة المشروع بغض النظر عما إذا كان حدث لها تغيير أم لا.

الأمر Web deployment Options بالقائمة Project

من خلال الأمر Web deployment Options يمكن تهيئة أدوات التحكم ActiveX أو الفورم التى من النوع ActiveForm التى تم الإنتهاء من إعدادها وهذه التهيئة بهدف جعل هذه العناصر متاحة للاستخدام فى خوادم شبكة الويب.

هذه التهيئة تنقسم إلى ثلاثة أقسام : قسم المشروع Project والذى يتم تهيئته من خلال التبويب Project بصندوق الحوار Web Deployment Options والموضح فى الشكل التالى :

شكل توضيحى :

Web Deployment Options
Project | Packages | Additional Files
Directories and URLs
Target dir (Full path of the deployed OCX):
Browse...
Target URL (Virtual path of the deployed OCX):
HTML dir (Full path of the deployed HTML file):
Browse...
General Options
Use CAB file compression
Include file version number
Auto increment release number
Deploy additional files
Default
OK | Cancel | Help

بهذا القسم يتحدد مواضع الملفات والعنوان URL لخادم الويب. كما إنه يسمح لك بأن تتعامل مع القيم التحديدية الخاصة بعملية التهيئة للملفات المضغوطة CAB والمعلومات الخاصة بالإصدار Version Information.

أما القسم الثانى فقسم الحزم البرمجية Packages التى يستخدمها المشروع والذى يـتم تهيئـتة من خلال التبويب Packages بصندوق الحوار Web Deployment Options كما هو موضح بالشكل التالى :

شكل توضيحى :

Web Deployment Options
Project | Packages | Additional Files
Packages used by this project
CAB Options
Use file VersionInfo
Directory and URL options
Target URL (or leave blank to assume file exists on target machines)
Target directory (or leave blank to assume file exists on server)
Browse...
Default　OK　Cancel　Help

وأخـيرا قسـم الملفـات الإضـافية المرتبطة بهذا المشروع والذى يتم تهيئتة من خلال التبويب Additional Files بصندوق الحوار Web Deployment Options كما هو موضح بالشكل التالى :

شكل توضيحى :

Web Deployment Options
Project | Packages | Additional Files
Files associated with project
CAB Options
Use file VersionInfo
Add
Directory and URL options
Target URL (or leave blank to assume file exists on target machines):
Target directory (or leave blank to assume file exists on server)
Browse...
Default　OK　Cancel　Help



الأمر Web Deploy بالقائمة Project

من خلال الأمر Web Deploy يمكن تهيئة أدوات التحكم ActiveX أو الفورم التى من النوع ActiveForm التى تم الإنتهاء من إعدادها وهذه التهيئة بهدف جعل هذه العناصر متاحة للاستخدام فى خوادم شبكة الويب.



هذا الأمر ينبغى استخدامه بعد استخدام الأمر Web deployment Options وإلا فلن يعمل.

الأمر Options بالقائمة Project

اختيار الأمر Options من القائمة Project يؤدى إلى فتح صندوق الحوار Project Options الموضح فى الشكل التالى :

شكل توضيحى :



هناك طريقة أخرى لفتح صندوق الحوار Project Options وهى الضغط على مجموعة المفاتيح Shift+Ctrl+F11 بلوحة المفاتيح.

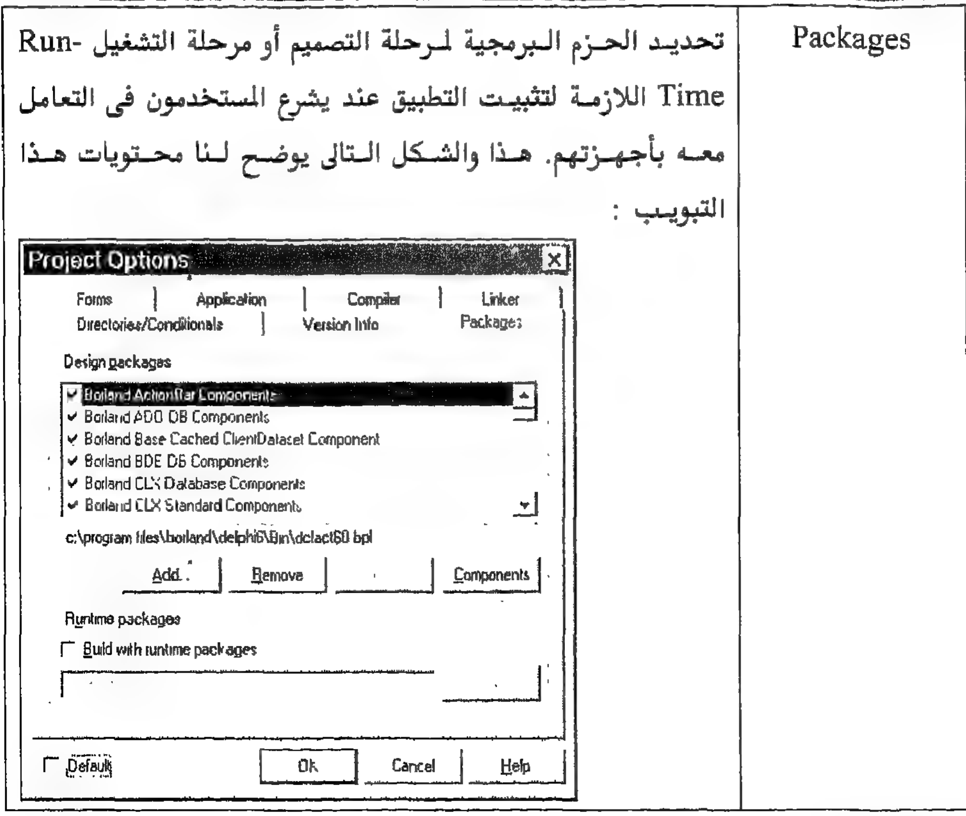
الجدول التالى يقدم لنا وصف مختصر لوظيفة كل تبويب من التبويبات الموجودة فى صندوق الحوار Project Options :



التبويب	الوصف والاستخدام
Forms	التحكم فى الفورم التى يتم إنشاؤها تلقائيا بواسطة المشروع. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :
Application	تحديد عنوان title للمشروع واسم ملف المساعدة واسم الأيقونة الملحقة بالمشروع. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :
Compiler	تحديد المحولات الخاصة بمترجم اللغة التى تتحكم فى طريقة ترجمة الكود البرمجى. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :

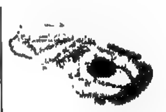
	Project Options Directories/Conditionals · Version Info · Packages · Forms · Application · Compiler · Linker Code generation: Optimization · Stack frames · Pentium-safe FDIV · 8 Record field alignment Syntax options: Strict var-strings · Complete boolean eval · Extended syntax · Typed @ operator · Open parameters · Huge strings · Assignable typed constants Runtime errors: Range checking · I/O checking · Overflow checking Debugging: Debug information · Local symbols · Reference info · Definitions only · Assertions · Use Debug DCUs Messages: Show hints · Show warnings Default · OK · Cancel · Help
Linker	مـن خـلال هذا التبويب يمكن التحكم فى الطريقة التى يتم بها ربط ملفـات المشـروع. هـذا والشـكل الـتالى يوضـح لـنا محـتويات هـذا التبويـب : Project Options Directories/Conditionals · Version Info · Packages · Forms · Application · Compiler · Linker Map file: Off · Segments · Publics · Detailed Linker output: Generate DCUs · Generate C object files · Generate C++ object files EXE and DLL options: Generate console application · Include TD32 debug info · Include remote debug symbols Memory sizes: Min stack size: $00004000 · Max stack size: $00100000 · Image base: $00400000 Description: EXE Description: Default · OK · Cancel · Help
Directories/Conditionals	تحديـد مواضـع الملفـات الضـرورية لترجمـة وربط المشروع الذى يتم العمـل بـه حالـيا. هـذا والشـكل الـتالى يوضـح لـنا محتويات هـذا التبويـب :

Version Info	تحديد نوعية المعلومات التى يتم إلحاقها بالمشروع ومنها معلومات عن رقم الإصدار وغيرها. هذا والشكل التالى يوضح لنا محتويات هذا التبويب :

Packages	تحديد الحزم البرمجية لمرحلة التصميم أو مرحلة التشغيل Run-Time اللازمة لتثبيت التطبيق عند يشرع المستخدمون فى التعامل معه بأجهزتهم. هذا والشكل التالى يوضح لنا محتويات هذا التبويب : 

القائمة Run

الأوامر الموجودة فى القائمة Run تساعدك فى اكتشاف الثغرات والأخطاء البرمجية الموجودة بالبرنامج الذى تعده. ليس هذا وفقط بل تساعدك أيضا فى معالجة هذه الأخطاء والثغرات.



الأوامر الموجودة فى القائمة Run هى المسئولة عن الأداء الوظيفى الجوهرى للأداة المتكاملة لفحص الأخطاء ومعالجتها Integrated Debugger.

الجدول التالى يقدم لنا وصف مختصر للأوامر الموجودة فى القائمة Run :

الوصف والاستخدام	الأمر
ترجمة وتنفيذ التطبيق الذى تعده. هذا ويمكن أيضا الضغط على المفتاح F9 بلوحة المفاتيح أو النقر بالماوس على الأيقونة .	Run
يعمل هذا الأمر على تقديم قائمة بالعمليات التى يتم تشغيلها حاليا والتى يمكنك فحصها لاكتشاف ما بها من أخطاء والعمل على معالجتها وذلك من خلال صندوق الحوار Attach to process الموضح فى الشكل التالى : Attach to Process Remote Machine: Running Processes: Image Name — PID — Path COMET.EXE — $FFE1B713 — C:\WINDOWS\SYSTEM\COMET\BIN\ DDHELP.EXE — $FFE142CF — C:\WINDOWS\SYSTEM\ DELPHI32.EXE — $FFE2133F — C:\PROGRAM FILES\BORLAND\DELPHI6\BIN\ DLATRAY.EXE — $FFE1B6F7 — D:\WIN98-PROGRAMS\VERITAS SOFTWARE\DLA\ EXPLORER.EXE — $FFFEB64F — C:\WINDOWS\ INTERNAT.EXE — $FFE13BBF — C:\WINDOWS\SYSTEM\ KERNEL32.DLL — $FFECDF8B — C:\WINDOWS\SYSTEM\ MDM.EXE — $FFFFFE73 — C:\WINDOWS\SYSTEM\ mmtask.tsk — $FFFF515B — C:\WINDOWS\SYSTEM\ MPREXE.EXE — $FFFF157F — C:\WINDOWS\SYSTEM\ MSTASK.EXE — $FFFFE51F — C:\WINDOWS\SYSTEM\ Show System Processes — Refresh — Cancel — Help هذا ويمكن أيضا استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .	Attach to Process
يعمل هذا الأمر على توصيف معاملات بداية التشغيل للتطبيق الذى تتولى إعداده كما يعمل على تحديد مضيف قابل للتنفيذ بالنسبة لملفات مكتبات الربط الديناميكى DLLs وذلك من خلال التبويب Local بصندوق الحوار Run Parameters الموضح فى الشكل التالى :	Parameters

	Run Parameters Local / Remote Host Application Browse... Parameters OK Cancel Help كما يمكن أيضا إجراء نفس ما سبق فى كمبيوتر من أجل إجراء عملية اكتشاف الأخطاء معالجتها من على بعد عن طريق الشبكات وذلك عن طريق التبويب Remote بصندوق الحوار Run Parameters الموضح فى الشكل التالى : Run Parameters Local / Remote Remote Remote Path: Remote Host: Parameters Debug project on remote machine OK Cancel Help
Register ActiveX Server	يعمل هذا الأمر على إضافة مدخل أحقية الاستخدام لبيئة الويندوز لأدوات التحكم ActiveX التى تعدها وهذا الأمر يكون متاح للاستخدام عندما يكون المشروع الحالى عبارة عن ActiveX Project. هذا وبعد الإنتهاء من استخدام هذا الأمر تظهر على الشاشة الرسالة الموضحة فى الشكل التالى :

	Information Successfully registered ActiveX Server, 'C:\Program Files\Borland\Delphi6\Projects\ActiveFormProj1.ocx' OK
Unregister ActiveX Server	يعمل هذا الأمر على إزالة المشروع من أحقية استخدام بيئة الويندوز. وهذا الأمر يكون متاح للاستخدام عندما يكون المشروع الحالى عبارة عن ActiveX Project. هذا وبعد الإنتهاء من استخدام هذا الأمر تظهر على الشاشة الرسالة الموضحة فى الشكل التالى : Information Successfully unregistered ActiveX Server, 'C:\Program Files\Borland\Delphi6\Projects\ActiveFormProj1.ocx' OK
Install COM+ Objects	من خلال هذا الأمر يتم تركيب وتثبيت كائنات فى المشروع الحالى داخل تطبيق من النوعية +COM. ويظهر هذا الأمر فى القائمة Run عندما يكون نظام التشغيل مدعما للتكنولوجيا +COM.
Step Over	من خلال هذا الأمر يتم تنفيذ الكود البرمجى سطر بسطر وفى أثناء ذلك يتم الإشارة إلى السطر الذى يتم تنفيذه حاليا بسهم أخضر فى نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى : Project1.dpr Unit1 Project1 begin Application.Initialize; Application.CreateForm(TForm1, Application.Run; end. 9 : 1 Modified Insert

الأمر	الوصف
	وهـذه العمليـة تتم عبر الإجراءات فى أثناء تنفيذهم كما لو كانوا وحـدة واحـدة. ونود هنا القول بأنه يمكن استدعاء هذا الأمر عن طريق الضغط على المفتاح F8 بلوحة المفاتيح.
Trace Into	يعمـل هذا الأمر على تنفيذ الكود البرمجى سطر بسطر مع إجراء عملـية تتـبع داخـل الإجـراءات بـنافذة محـرر الكـود الـبرمجى Code Editor كما هو موضح بالشكل التالى : Unit1.pas TForm1 Variables/Constants Uses Unit1 Project1 implementation {$R *.dfm} end. 24: 1 Modified Insert هـذا ويمكـن اسـتدعاء الأمـر عـن طـريق الضـغط على المفتاح F7 بلوحة المفاتيح.
Trace To Next Source Line	يعمـل هـذا الأمـر يقوم بتنفيذ الكود البرمجى وفى أثناء ذلك يتم الـتوقف عـند السـطر الـتالى القـابل للتنفـيذ بـنافذة محـرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى : Project1.dpr Uses Unit1 Project1 {$R *.res} begin Application.Initialize; Application.CreateForm(TForm1, Application.Run; end. 10: 1 Modified Insert

الأمر	الوصف
	هـذا ويمكـن اسـتدعاء الأمـر عـن طـريق الضـغط عـلى المفـتاحين Shift+F7 بلوحة المفاتيح.
Run To Cursor	يقـوم هـذا الأمـر بتشـغيل البرنامج المحمل حتى يتم الوصول إلى موضع مؤشر الكتابة فى الكود البرمجى داخل نافذة محرر الكود الـبرمجى Code Editor. هـذا ويمكـن اسـتدعاء الأمر عن طريق الضغط على المفتاح F4 بلوحة المفاتيح.
Run Until Return	يقوم هذا الأمر بتشغيل البرنامج حتى يتم الحصول على أى مخرجات مـن الدالـة التى يتم التعامل معها حاليا. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاحين Shift+F8 بلوحة المفاتيح.
Show Execution Point	يعمـل هـذا الأمـر عـلى وضع مؤشر الكتابة عند نقطة التنفيذ فى النافذة CPU كما هو موضح بالشكل التالى :
Program Pause	من خلال هذا الأمر يمكن تعليق تنفيذ البرنامج بشكل مؤقت.
Program Reset	يقوم هذا الأمر بإنهاء تشغيل التطبيق وإزالته من ذاكرة الكمبيوتر العشوائية وجعل التطبيق ينتقل من مرحلة التشغيل Run-Time إلى مود التصميم. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاحين Ctrl+F2 بلوحة المفاتيح.

الأمر	الوصف
Inspect	يعمل هذا الأمر على فتح صندوق الحوار Inspector الموضح فى الشكل التالى : من خلال صندوق الحوار هذا تستطيع إدخال أى عنصر ترغب فى فحصه.
Evaluate/Modify	يعمل هذا الأمر على عرض النافذة Evaluate/Modify الموضحة فى الشكل التالى : فى هذه النافذة تستطيع أن تقوم بتقييم أو تعديل القيمة الخاصة بأى تعبير موجود فى الكود البرمجى ويتم تنفيذه حاليا. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاحين Ctrl+F7 بلوحة المفاتيح.
Add Watch	يعمل هذا الأمر على فتح صندوق حوار خصائص المراقبة Watch Properties الموضح فى الشكل التالى :

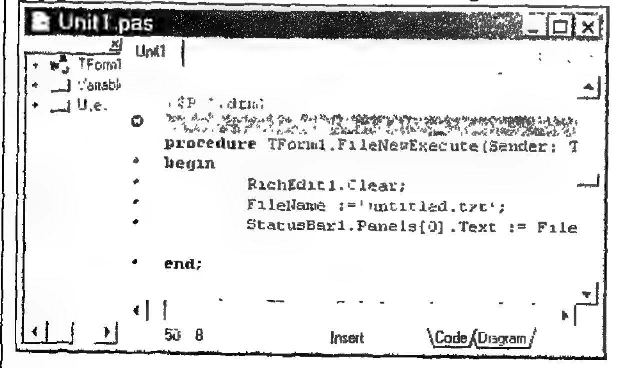
	Watch Properties Expression: end Repeat count: 0 — Digits: 18 ☑ Enabled — ☐ Allow Function Calls ○ Character ○ Hexadecimal ○ Record/Structure ○ String ○ Floating point ● Default ○ Decimal ○ Pointer ○ Memory Dump OK — Cancel — Help بصندوق الحوار هذا تستطيع أن إنشاء وتعديل أدوات المراقبة التى يتم استخدامها فى أثناء متابعة تنفيذ الكود البرمجى. هذا ويمكن استدعاء الأمر عن طريق الضغط على المفتاحين Ctrl+F5 بلوحة المفاتيح.
Add Breakpoint	لهذا الأمر قائمة فرعية تشتمل على مجموعة من الأوامر التى يمكن من خلالها إنشاء وتعديل نقط التوقف بالكود البرمجى أثناء تنفيذه فى مرحلة التشغيل Run-Time.

الجدول التالى يقدم لنا وصف مختصر لوظيفة كل أمر من الأوامر الموجودة بالقائمة الفرعية الخاصة بالأمر Add Breakpoint :

الأمر	الوصف والاستخدام
Source Breakpoint	يعمل هذا الأمر على فتح صندوق الحوار Add Source Breakpoint الموضح فى الشكل التالى : Add Source Breakpoint Filename: \Delphi6\Projects\TextEditor\Unit1.pas Line number: 72 Condition: Pass count: 0 Group: Advanced >> OK — Cancel — Help



بصندوق الحوار هذا تستطيع أن تضع نقطة توقف فى سطر معين بالكود البرمجى المصدرى بالتطبيق الذى تعمل به. ونتيجة ذلك أنه عندما تقوم بتشغيل التطبيق تجد أن نقطة التنفيذ فى محرر الكود البرمجى Code Editor تشير إلى موضع نقطة التوقف كما هو موضح بالشكل التالى :



هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .

Address Breakpoint

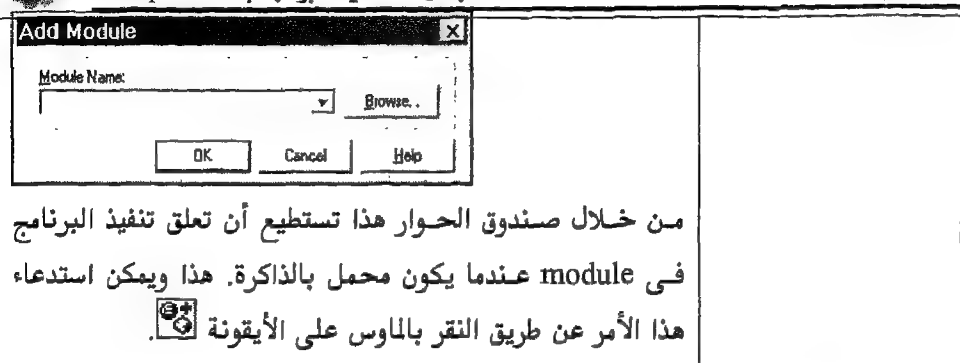
يعمل هذا الأمر على فتح صندوق الحوار Add Address Breakpoint الموضح فى الشكل التالى :

Add Address Breakpoint

Address: 50
Condition:
Pass count: 0
Group:
Advanced >>
OK | Cancel | Help

بصندوق الحوار هذا تستطيع أن تضع نقطة توقف فى آلية معينة من آليات تنفيذ الأوامر machine instruction. وعندما تقوم بتشغيل التطبيق نجد أن نقطة التنفيذ فى النافذة CPU تشير إلى موضع نقطة التوقف كما هو موضح بالشكل التالى :

	هـذا ويمكـن اسـتدعاء هـذا الأمـر عـن طـريق الـنقر بالماوس على الأيقونة .
Data Breakpoint	عـن طـريق هـذا الأمـر يـتم فـتح صـندوق الحـوار Add Data Breakpoint الموضح فى الشكل التالى : بصـندوق الحـوار هذا تستطيع وضع نقطة توقف فى عنوان معين لكـى تعمـل عـلى إيقاف تنفيذ البرنامج عندما تصل نقطة التنفيذ إلى الموضـع الموجـودة فيه نقطة التوقف. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .
Module Load Breakpoint	عن طريق هذا الأمر يتم فتح صندوق الحوار Add/Edit Module الموضح فى الشكل التالى :



مـن خـلال صـندوق الحـوار هذا تستطيع أن تعلق تنفيذ البرنامج فـى module عـندما يكون محمل بالذاكرة. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .

خطـوات إضـافة نقـط توقـف مـن خـلال مشـاهد اكتشـاف ومعالجـة الأخطـاء debugging views

(١) قـم أولا بتشـغيل التطبيق الذى تعده. ثم افتح القائمة View واختر منها الأمر Debug Windows ثم اختر Modules لتظهر على الشاشة النافذة Module List الموضحة فى الشكل التالى :

شكل توضيحى :

(٢) انقـر بالـزر الأيمـن عـلى أى Module من الـModules الموجودة فى الجانب الأيسـر العلوى من النافذة Modules List ثم اختر Add module من القائمة المختصرة التى تظهر على الشاشة كما هو موضح بالشكل التالى :

شكل توضيحى :

القائمة Component

تشتمل القائمة Component مجموعة الأوامر التالية :

الأمر	الوصف والاستخدام
New Component	يقوم هذا الأمر بفتح صندوق الحوار New Component الموضح فى الشكل التالى : بصندوق الحوار هذا تستطيع إنشاء مكون جديد من خلال تحديد كل من نوع الكائن الذى سينحدر منه واسم القطاع الذى سيوجد به واختيار الباللية التى سيوضع بها بباللية المكونات وتحديد اسم ملف الوحدة الخاص به وتحديد أين سيتم تخزينه. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .

<table>
<tr>
<td>Install Component</td>
<td>يقوم هذا الأمر بتركيب وتثبيت مكون داخل حزمة برمجية موجودة بالفعل وذلك من خلال التبويب Into Existing Package بصندوق الحوار Install Component الموضح فى الشكل التالى :

Install Component
Into existing package | Into new package
Unit file name — Browse...
Search path — $(DELPHI)\Lib;$(DELPHI)\Bin;$(DELPHI)\Imports;$(DELPHI)\Projects\Bpl
Package file name: — c:\program files\borland\delphi6\Lib\dclusr.dpk — Browse...
Package description — Borland User Components
OK — Cancel — Help

كما يمكن أيضا تركيب وتثبيت مكون داخل حزمة برمجية جديدة بالفعل وذلك من خلال التبويب Into New Package بصندوق الحوار Install Component الموضح فى الشكل التالى :

Install Component
Into existing package | Into new package
Unit file name: — Browse
Search path — $(DELPHI)\Lib;$(DELPHI)\Bin;$(DELPHI)\Imports;$(DELPHI)\Projects\Bpl
Package file name: — Browse...
Package description
OK — Cancel — Help

هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .</td>
</tr>
<tr>
<td>Import ActiveX Control</td>
<td>يعمل هذا الأمر على إضافة مكتبات الأنواع الخاصة بأدوات التحكم إلى المشروع الذى تعده وهذه الإضافة تتم عن طريق صندوق الحوار Import ActiveX الموضح فى الشكل التالى :</td>
</tr>
</table>

	Import ActiveX Import ActiveX VideoSoft vsFlex3 Controls (Version 3.0) VideoSoft vsOcx Controls (Version 5.0) AcDcToday ActiveX Control module (Version 1.0) AcPreview ActiveX Control module (Version 1.0) Acrobat Control for ActiveX (Version 1.3) Active Setup Control Library (Version 1.0) ActiveBar Control (Version 1.0) Add... Class names: Palette page: ActiveX Unit dir name: C:\Program Files\Borland\Delphi6\Imports Search path: $(DELPHI)\Lib;$(DELPHI)\Bin;$(DELPHI)\Impo Cancel Help هـذا ويمكـن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة
Create Component Template	مـن خـلال هـذا الأمـر يمكـن تفصيل مكونات وحفظها على أساس كونهـا قوالـب بأسمـاء جديـدة وفى صفحة جديدة بباليتة المكونات وبأيقونات خاصة بهم أيضا.
Install Packages	مـن خـلال هـذا الأمـر يـتم فـتح صندوق الحوار Project Options الموضح فى الشكل التالى : Project Options Packages Design packages Borland ActionBar Components Borland ADO DB Components Borland Base Cached ClientDataset Component Borland BDE DB Components Borland CLX Database Components Borland CLX Standard Components c:\program files\borland\delphi6\Bin\dclact60.bpl Add... Remove Edit Components Runtime packages Build with runtime packages Default OK Cancel Help

	بصندوق الحوار هذا يتم تحديد الحزم البرمجية التي يحتاج إليها المشروع الذى تعده. هذا ويمكن استدعاء هذا الأمر عن طريق النقر بالماوس على الأيقونة .
Configure Palette	يعمل هذا الأمر على فتح صندوق الحوار Palette Properties الموضح فى الشكل التالى : من خلال صندوق الحوار هذا يمكن إضافة مكونات جديدة إلى بالليتة المكونات كما يمكن أيضا إزالة أى مكونات منها وكذلك تغيير اسماء أى من المكونات الموجودة بها بالإضافة إلى إمكانية تغيير ترتيب أيقونات المكونات فى كل صفحة من صفحات بالليتة المكونات.

القائمة Database

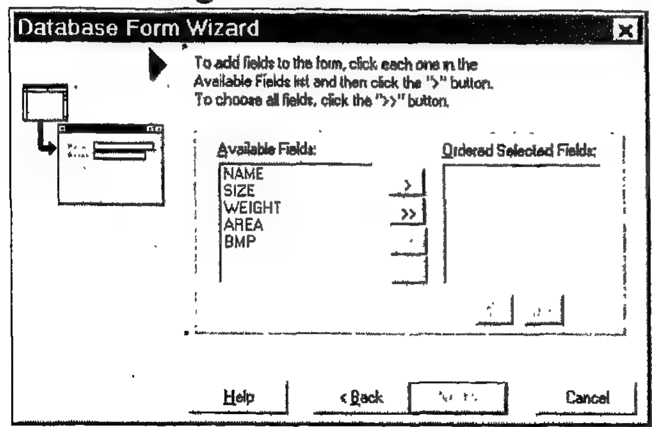
الأوامر الموجودة فى القائمة Database تمكنك من إنشاء وتعديل وتتبع ومشاهدة قواعد البيانات. هذا والجدول التالى يقدم لنا وصفا مختصرا لوظيفة الأوامر الموجودة بهذه القائمة :

الأمر	الوصف والاستخدام
Explorer	يعمل هذا الأمر على فتح النافذة SQL Explorer الموضحة فى الشكل التالى :

	كما يمكن أن يتم فتح النافذة Database Explorer وذلك إعتمادا على نوع إصدار اللغة الذى تتعامل معه. وعلى العموم فكلا النافذتين تسمح لك بأن تنشأ وتشاهد وتحرير البيانات والـBDE aliases. كما أن الأداة SQL Explorer تسمح لك بأن تعد أدوات استفسار لكل من قواعد البيانات المحلية أو البعيدة.
SQL Monitor	عند استخدام هذا الأمر يتم فتح النافذة SQL Monitor الموضحة فى الشكل التالى : هذه الأداة تكون متاحة للاستخدام فقط فى بعض إصدارات لغة

	Delphi وهى تسمح لك بأن تراقب SQL resource allocation ومشاهدة جمل الاستدعاءات الحقيقية التى يتم التعامل معها من خلال أدوات الربط SQL Links وهذه الإستدعاءات تكون لخادم بعيد أو لمصدر من مصادر البيانات ODBC وذلك من خلال ODBC Socket.
Form Wizard	عند اختيار هذا الأمر يتم فتح أول صندوق حوار فى المعالج Database Form Wizard (المستخدم لإنشاء فورمة تعرض بيانات من قواعد بيانات محلية أو بعيدة) الموضح فى الشكل التالى : Database Form Wizard What type of form do you want the wizard to create? Form Options: Create a simple form / Create a master/detail form DataSet Options: Create a form using TTable objects / Create a form using TQuery objects Help · Next > · Cancel صندوق الحوار هذا يطلب تحديد نوعية الفورمة التى تود إنشاؤها. بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى : Database Form Wizard Choose a table to use with the form. Table Name: *.DBF;*.DB · Directories: c:\...\borland\delphi6\projects C:\ · Program Files · Borland · Delphi6 · Projects · are List Files of Type: All Tables (*.DBF;*.DB) · Drive or Alias name: c: [shenl] Help · < Back · Cancel

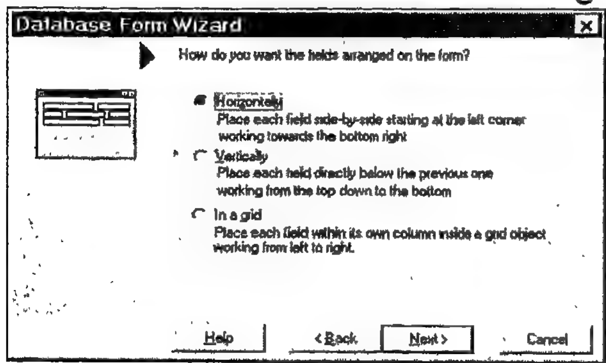
بصندوق الحوار هذا تستطيع اختيار الجدول الذى سيكون مصدر للبيانات التى سيتم عرضها فى الفورمة. بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى:



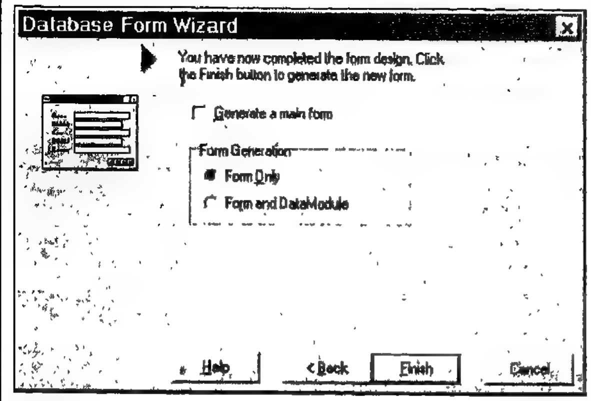
بصندوق الحوار هذا تستطيع اختيار الحقول التى سيتم وضعها فى الفورمة وللقيام بذلك علم بالماوس على الحقول التى ترغبها فى قائمة العرض Available Fields ثم انقر بالماوس على الأيقونة [>] (أو انقر بالماوس على الأيقونة [>>] لنقل الحقول كلها من قائمة العرض Available Fields إلى قائمة العرض Ordered Selected Fields كما هو موضح بالشكل التالى :

Database Form Wizard
To add fields to the form, click each one in the Available Fields list and then click the ">" button.
To choose all fields, click the ">>" button.
Available Fields:
Ordered Selected Fields:
NAME
SIZE
WEIGHT
AREA
BMP
Help
< Back
Next >
Cancel

والآن انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى :



صندوق الحوار هذا يطلب منك تحديد طريقة ترتيب الحقول المختارة فى الفورمة (أفقيا أو رأسيا أو فى مجموعة خلايا كما فى برامج الجداول الإلكترونية مثل برنامج Excel). بعد ذلك انقر بالماوس على المفتاح Next للانتقال إلى صندوق الحوار التالى والموضح بالشكل التالى :



يعتبر صندوق الحوار هذا آخر صندوق حوار فى المعالج Database Form Wizard وهو يطلب منك أن تنقر بالماوس على المفتاح Finish للبدء فى تكوين الفورمة والتى يمكن أن تكون مشابهه للفورمة الموضحة فى الشكل التالى :

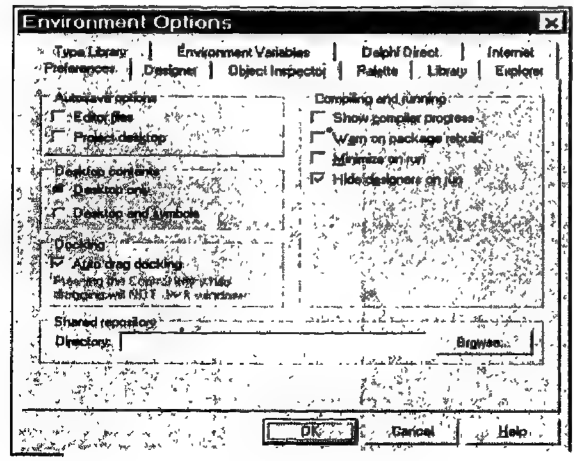
القائمة Tools

يمكن استخدام قائمة الأدوات Tools للقيام بالآتى :

- مشاهدة القيم التحديدية التى تتحكم فى خصائص بيئة العمل مع إمكانية تغيير أى من هذه القيم التحديدية.
- مشاهدة القيم التحديدية التى تتحكم فى خصائص عملية التحرير Editing مع إمكانية تغيير أى من هذه القيم التحديدية.
- مشاهدة القيم التحديدية التى تتحكم فى خصائص وصفات أداة اكتشاف ومعالجة الأخطاء والثغرات Debugger مع إمكانية تغيير أى من هذه القيم التحديدية.
- التعديل فى العناصر الموجودة فى مستودع الكائنات Object Repository.
- التعديل فى مجموعة البرامج الموجودة فى القائمة Tools.
- إنشاء وتعديل جداول قواعد البيانات المحلية.
- إنشاء وتعديل مجموعات الحزم البرمجية.
- إنشاء وتحرير الصور.

فيما يلى سنستعرض سويا من خلال الجدول التالى الأوامر الأساسية Default الموجودة فى القائمة Tools :

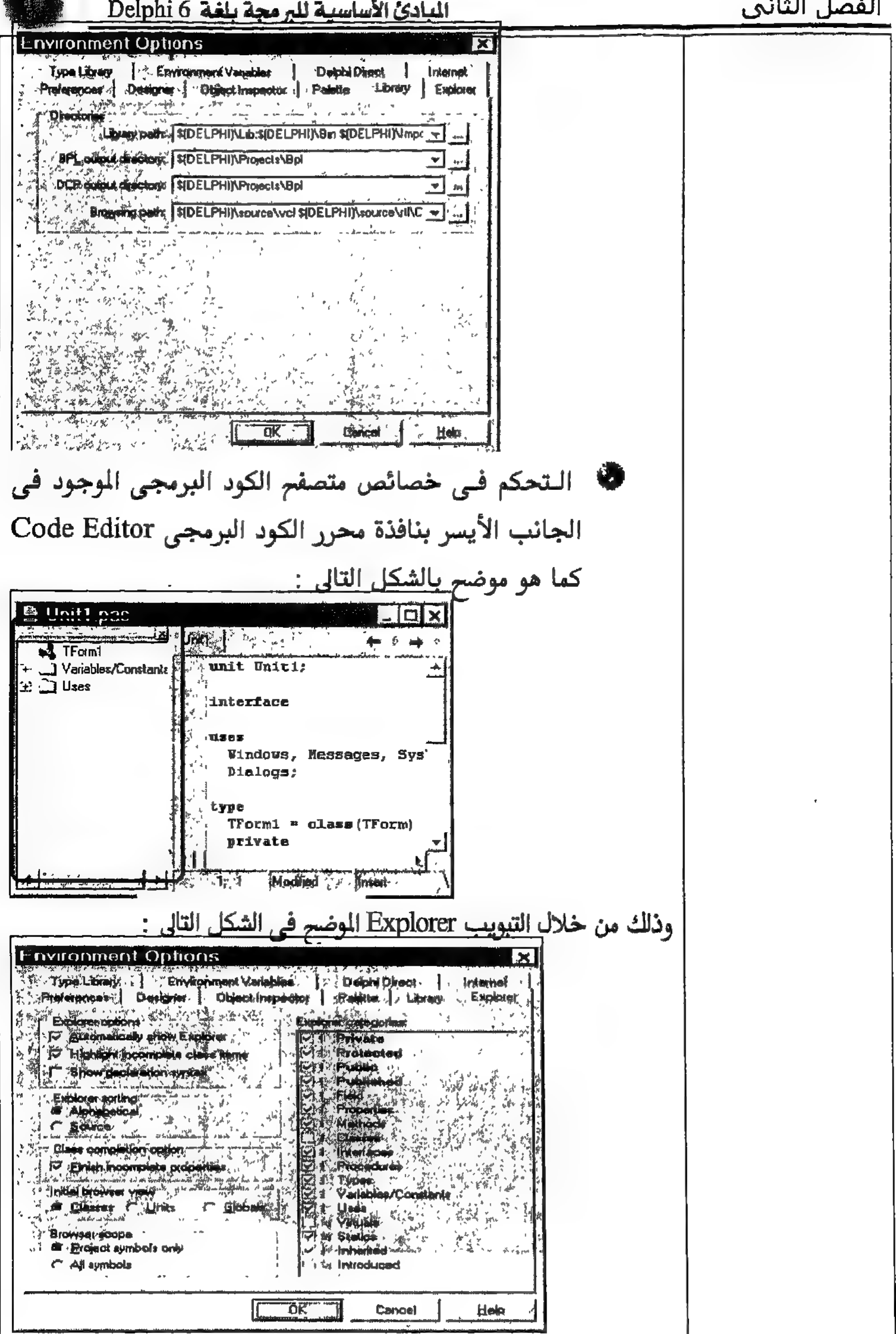


الأمر	الوصف والاستخدام
Environment Options	مـن خـلال هـذا الأمـر يـتم عرض صندوق حوار خيارات بيئة العمل Environment Options الموضح فى الشكل التالى :  عن طريق صندوق الحوار هذا تستطيع أن تقوم بكل من الآتى : • تحديـد القـيم المفضـلة لتهيـئة بيـئة العمل وذلك من خلال التبويب Preferences الموضح فى الشكل السابق. • تحديـد القـيم المفضـلة لأداة تصـميم الفورم وذلك من خلال التبويب Designer الموضح فى الشكل التالى :

- الـتحكم فـى خصـائص النافذة Object Inspector وذلك مـن خلال التبويب Object Inspector الموضح فى الشكل التالى :

- الـتحكم فـى خصـائص باللبـتة المكونـات وذلك من خلال التبويب Palette الموضح فى الشكل التالى :

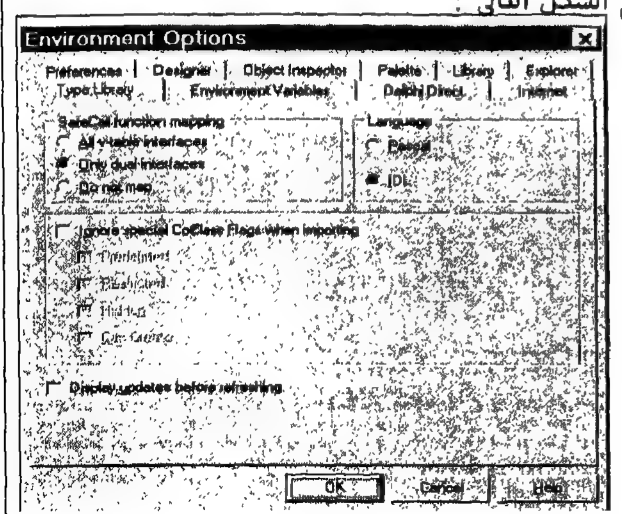
- الـتحكم فى خصائص كل من مترجم اللغة وأداة الربط Linker والفهـارس الخاصـة بهمـا لجمـيع الحـزم الـبرمجية وذلك من خلال التبويب Library الموضح فى الشكل التالـى :



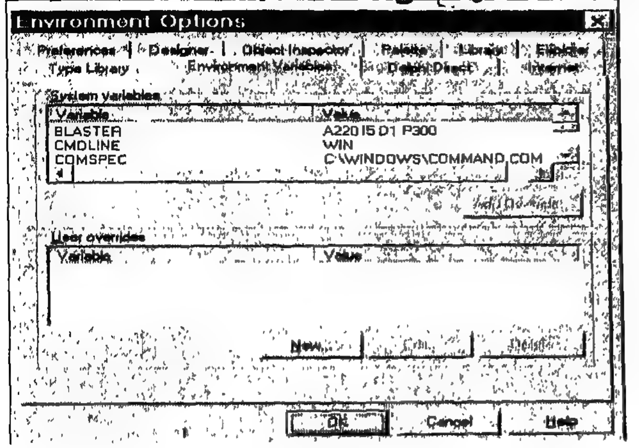
- الـتحكم فـى خصائص متصفح الكود البرمجي الموجود فى الجانب الأيسر بنافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

وذلك من خلال التبويب Explorer الموضح فى الشكل التالى :

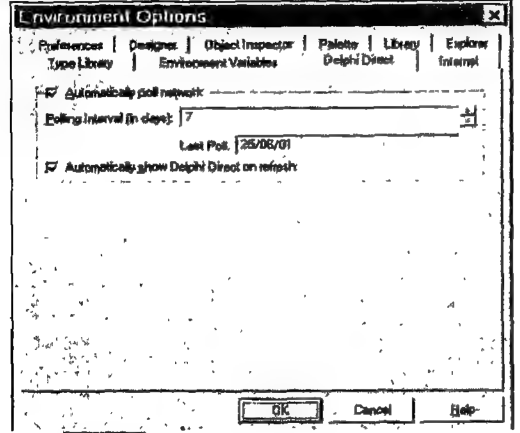
- التحكم فى خصائص مكتبات الأنواع والمسارات الدالة على وجودها وذلك من خلال التبويب Type Library الموضح فى الشكل التالى :



- التحكم فى القيم الخاصة بالمتغيرات التى تحدد صفات بيئة العمل وذلك من خلال التبويب Environment Variables الموضح فى الشكل التالى :



- التحكم فى إمكانية الوصول إلى المصادر المتاحة بشبكة الإنترنت للغة Delphi وذلك من خلال التبويب Delphi Direct الموضح فى الشكل التالى :

	 ● التحكم فى خصائص إمكانية تفاعل لغة Delphi مع الملفات المعدة بلغات البرمجة للإنترنت مثل لغة XML ولغة HTML وذلك من خلال التبويب Internet الموضح فى الشكل التالى :
Editor Options	يعمل هذا الأمر على فتح صندوق الحوار Editor Properties الذى يمكن من خلاله تحديد القيم المفضلة لتهيئة محرر الكود البرمجى Code Editor. وهذه التهيئة تتم من خلال الآتى : التحكم فى الخصائص العامة لمحرر الكود البرمجى وذلك عن طريق التبويب General الموضح فى الشكل التالى :

Editor Properties

General | Display | Key Mappings | Color | Code Insight

Editor options

- [x] Auto indent mode
- [x] Insert mode
- [ ] Use tab character
- [ ] Smart tab
- [ ] Optimal fill
- [x] Backspace unindents
- [x] Cursor through tabs
- [x] Group undo
- [ ] Cursor beyond EOF
- [ ] Undo after save
- [ ] Keep trailing blanks
- [ ] BRIEF regular expressions
- [ ] Persistent blocks
- [x] Overwrite blocks
- [ ] Double click line
- [x] Find text at cursor
- [ ] Force cut and copy enabled
- [x] Use syntax highlight

Editor SpeedSetting: Default keymapping

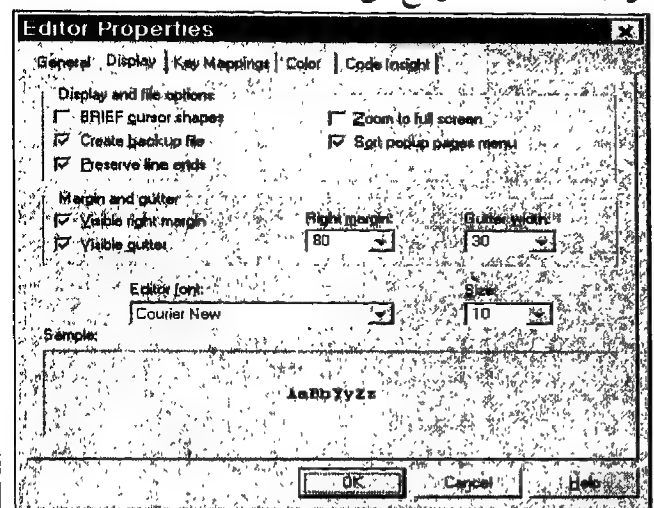
Block indent: 2 Undo limit: 32767

Tab stops: 8

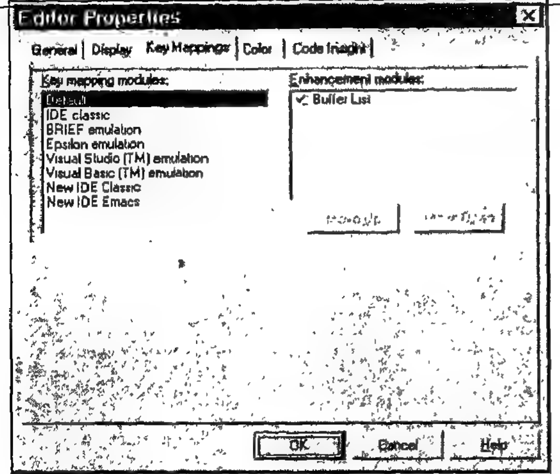
Syntax extensions: pas;dpr;dpk;inc;dfm;xfm

OK Cancel Help

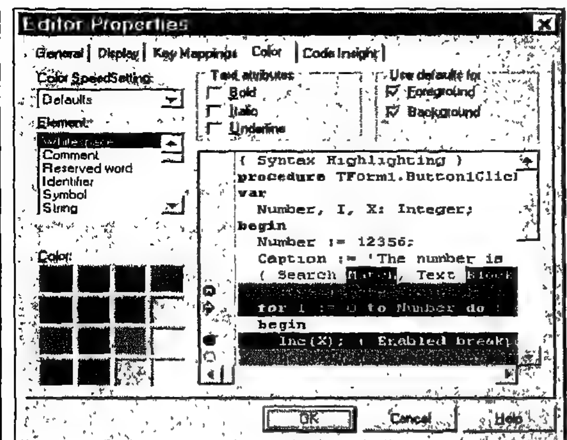
الـتحكم فى طريقة عرض النصوص بمحرر الكود البرمجى وذلك عن طريق التبويب Display الموضح فى الشكل التالى :



التحكم فى خصائص key mapping Modules مع إمكانية تمكين أو تعطيل أى منها بمحرر الكود البرمجى وذلك عن طريق التبويب Key Mapping الموضح فى الشكل التالى :



التحكم فى طريقة تلوين أجزاء معينة فى الكود البرمجى مثل الكلمات المحجوزة وأسماء الدوال...وذلك عن طريق التبويب Color الموضح فى الشكل التالى :



التحكم فى طريقة مشاهدة الكود البرمجى بمحرر الكود البرمجى وفى إمكانية الاستكمال التلقائى للجمل فى أثناء كتابتها وذلك عن طريق التبويب Code Insight الموضح فى الشكل التالى :

	Editor Properties General | Display | Key Mappings | Color | Code Insight Automatic features ☑ Code completion ☑ Code parameters ☑ Tooltip expression evaluation ☑ Tooltip symbol insight Delay: 0.1 sec 1.5 sec Code templates Templates: Name | Description arrayd | array declaration (var) arrayc | array declaration (const) Add... Edit... Delete Code: array[0..] of ; OK Cancel Help
Debugger Options	يعمل هذا الأمر على فتح صندوق الحوار Debugger Options والذى يمكن من خلاله التحكم فى خصائص وصفات أداة اكتشاف ومعالجة الأخطاء والثغرات وهذا التحكم يتم من خلال التبويبات الأربعة التالية : التبويب General الموضح فى الشكل التالى : Debugger Options General | Event Log | Language Exceptions | OS Exceptions General ☐ Map TD32 keystrokes on run ☐ Allow function calls in new watches ☐ Mark buffers read-only on run ☐ Rearrange editor local menu on run ☐ Inspectors stay on top ☐ Debug spawned processes Inspector Defaults ☑ Show inherited ☐ Show fully qualified names Paths Debug Symbols Search Path: Debug DCU Path: $(DELPHI)\Lib\Debug ☑ Integrated debugging OK Cancel Help التبويب Event Log الموضح فى الشكل التالى :

Debugger Options
General | Event Log | Language Exceptions | OS Exceptions |
General
Clear log on run
Unlimited length
Length:
Display process info with event
Messages
Breakpoint messages
Process messages
Thread Messages
Output messages
Window messages
Integrated debugging
OK | Cancel | Help

التبويب Language Exceptions الموضح فى الشكل التالى :

Debugger Options
General | Event Log | Language Exceptions | OS Exceptions |
Exception Types to Ignore
Delphi EAbort Exceptions
Microsoft DAO Exceptions
VisiBroker Internal Exceptions
CORBA System Exceptions
CORBA User Exceptions
Add
Stop on Delphi Exceptions
Integrated debugging
OK | Cancel | Help

التبويب OS Exceptions الموضح فى الشكل التالى :

Debugger Options
General | Event Log | Language Exceptions | OS Exceptions |
Exceptions
Access Violation ($C0000005)
In Page Error ($C0000006)
Invalid Handle ($C0000008)
No Memory ($C0000017)
Illegal Instruction ($C000001D)
Noncontinuable Exception ($C0000025)
Invalid Disposition ($C0000026)
Array Bounds Exceeded ($C000008C)
Float Denormal Operation ($C000008D)
Float Divide By Zero ($C000008E)
Float Inexact Result ($C000008F)
Float Invalid Operation ($C0000090)
Handled by
Debugger
User program
On resume
Run handled
Run unhandled
Add
Integrated debugging
OK | Cancel | Help

Repository	يعمل هذا الأمر على فتح صندوق الحوار Object Repository الموضح فى الشكل التالى : من خلال صندوق الحوار هذا يمكن إضافة صفحات جديدة إلى مستودع الكائنات كما يمكن أيضا مسح أى صفحة من الصفحات الموجودة بالفعل كما نستطيع أيضا تغيير اسم أى من هذه الصفحات
Translation Tools Options	عن طريق هذا الأمر يتم فتح صندوق الحوار Translation Tools Options والذى يمكن من خلاله تهيئة وإعداد أدوات الترجمة اللغوية Translation Tools وذلك من خلال التبويبات الثلاثة الموجودة به : التبويب Preferences الموضح فى الشكل التالى :

	التبويب Colors الموضح فى الشكل التالى : التبويب Fonts الموضح فى الشكل التالى :
Translation Repository	يعمل هذا الأمر على فتح نافذة مستودع الترجمة Translation Repository الموضح فى الشكل التالى :

	Translation Repository New Open Save Save As Import Help Exit مـن خـلال هـذه الـنافذة يمكـن تخـزين واستعادة السلاسل الحرفية المترجمة فى مدير الترجمة.
External Editor	عـن طريق هـذا الأمـر يـتم الولـوج إلى محـرر صـفحات الويب وهذا المحـرر ينـتمى إلى الطائفـة third-party وهـذا المحـرر يـتألف من التبويبات الثلاثة التالية : التبويب ActiveFormProj1 الموضح فى الشكل التالى : ActiveFormProj1 dpr ActiveFormProj1 | ActiveFormImpl1 | deploy.txt library ActiveFormProj1; uses ComServ, ActiveFormProj1_TLB in 'ActiveFormProj1_TLB.pas', ActiveFormImpl1 in 'ActiveFormImpl1.pas' {ActiveForm {$E ocx} exports DllGetClassObject, DllCanUnloadNow, DllRegisterServer, DllUnregisterServer; 1: 1 Modified Insert Code التبويب ActiveFormImpl1 الموضح فى الشكل التالى :

	ActiveFormImpl1.pas ActiveFormProj1 ActiveFormImpl1 deploy.txt unit ActiveFormImpl1; {$WARN SYMBOL_PLATFORM OFF} interface uses Windows, Messages, SysUtils, Classes, Graphics, Cont ActiveX, AxCtrls, ActiveFormProj1_TLB, StdVcl; type TActiveFormX = class(TActiveForm, IActiveFormX) private { Private declarations } FEvents: IActiveFormXEvents; 1: 1 Modified Insert Code Diagram التبويب Deploy.txt الموضح فى الشكل التالى : C:\Program Files\Borland\Delphi6\deploy.txt ActiveFormProj1 ActiveFormImpl1 deploy.txt Deploying Delphi Applications Please read this entire file before deploying Delphi 6 lications, components, or libraries. This file is prov pursuant to the Borland No-Nonsense License Statement provides information on deploying Delphi 6 application Table of Contents 1: 1 Insert Code Preview
Web App Debugger	يعمـل هـذا الأمـر عـلى فتح النافذة Web App Debugger الموضح فى الشكل التالى : Web App Debugger Server Help Start Port 1024 Default URL: http://localhost:1024/ServerInfo.ServerInfo Statistics Log RequestCount: 0 Total Response Time: 0 Avg Response Time: 0 Last Response Time: 0 Min Response Time: 0 Max Response Time: 0



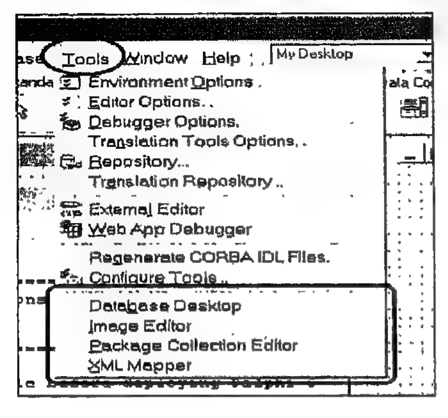
	عن طريق هذه النافذة تستطيع أن تختبر تطبيقات التى تعدها للعمل فـى خـوادم شـبكة الويـب كمـا تسـتطيع أيضـا اكتشـاف ومعالجـة الأخطاء والثغرات التى تظهر بها.
Regenerate CORBA IDL Files	عـن طـريق هـذا الأمـر يـتم فتح صندوق الحوار Regenerate IDL Files والـذى يمكـن مـن خلالـه تكويـن تطبـيق client أو Server مبـنى عـلى الملفـات الـتى مـن النوعـية IDL Files. هـذا ويـتألف صندوق الحوار هذا من التبويبات التالية : التبويب Application الموضح فى الشكل التالى : التبويب Options الموضح فى الشكل التالى :

الأمر	الوصف
Configure Tools	يعمل هذا الأمر على فتح صندوق الحوار Tool Options الموضح فى الشكل التالى : يمكـن استخدام صندوق الحوار هذا لإضافة أو إزالة أو تعديل الأوامر الموجودة فى القائمة Tools.

الخيارات الإضافية بالقائمة Tools

الجزء السفلى بالقائمة Tools والموضح فى الشكل التالى يمكن تفصيله كما يـروق لك :

شكل توضيحى :



فأنـت تسـتطيع إزالـة الأدوات الموجـودة فـى هـذا الجزء كما تستطيع أيضا إضافة أدوات أخـرى من الأدوات التى سبق تثبيتها أو تفصيلها والتى ترغب فى التعامل معها فى أثناء استخدام اللغة.

الجـدول الـتالى يقـدم لنا وصف مختصر للأوامر الإفتراضية Default الموجودة بهذا الجزء من القائمة Tools :

الأمر	الوصف والاستخدام
Database Desktop	عـند اختيار هذا الأمر يتم فتح النافذة Database Desktop الموضح فى الشكل التالى : وهذه النافذة تمثل أداة يمكن من خلالها إنشاء أو إعادة بناء جداول قواعـد البيانات كما يمكن أيضا استعراض البيانات المخزنة فى هذه الجـداول والـتعديل بهـا أيضـا. وأنت تستطيع التعامل مع الجداول الموجـودة بقواعد البيانات المصممة من خلال Paradox وdBASE و SQL. هـذا ونـود هـنا القـول بـأن هـذه الأداة توجد فى الأصل فى المسار التالى : Program Files\Common Files\Borland Shared\Database Desktop
Image Editor	مـن خـلال هـذا الأمـر يمكـن التعامل مع أداة لإنشاء وتحرير الصور الـتى يتم استخدامها فى التطبيق الذى تعده. وهذه الأداة تتمثل فى النافذة Image Editor الموضحة فى الشكل التالى :



Package Collection Editor	يعمل هذا الأمر على فتح النافذة Package Collection Editor الموضحة فى الشكل التالى : هذه النافذة تسمح لك بأن تشاهد وتحرر الحزم البرمجية والملفات الأخرى الملحقة بأى من مجموعات الحزم البرمجية وبهذه الطريقة يمكن توزيع الحزم البرمجية لكى يستخدمها المبرمجون الآخرون.
XML Mapper	عند اختيار هذا الأمر يتم فتح النافذة XML Mapper Tool الموضحة فى الشكل التالى :

من خلال هذه النافذة تستطيع أن تقوم بتعريف الـmappings التى تتم بين المستندات المعدة بلغة XML وحزم البيانات التى تستخدمها مجموعات البيانات العميلة client datasets.

الفصل الثالث

استخدام
مكتبة المكونات Components
لدى Delphi 6

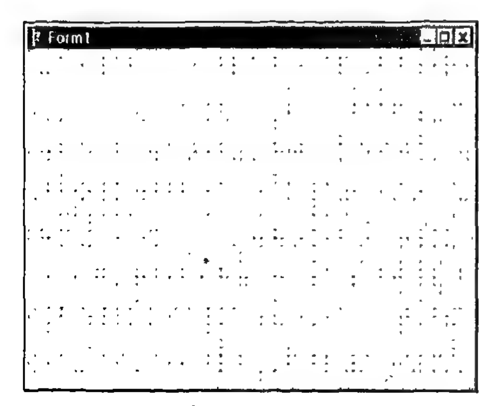


بيئة التطوير المتكاملة IDE

عندما تبدأ في تشغيل لغة Delphi فإنك تتواجد بشكل مباشر داخل بيئة التطوير المتكاملة IDE الخاصة بلغة Delphi. وبيئة العمل هذه تعمل على توفير كافة الأدوات التى تحتاج لها لتصميم وتطوير واختبار وفحص Debug ونشر وتوزيع التطبيقات.

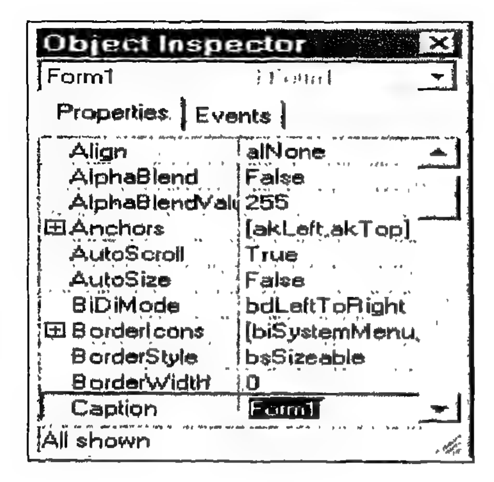
تتضمن بيئة التطوير المتكاملة IDE الخاصة بلغة Delphi على أداة ظاهرية لتصميم الفورم والموضحة بالشكل التالى :

شكل توضيحى :



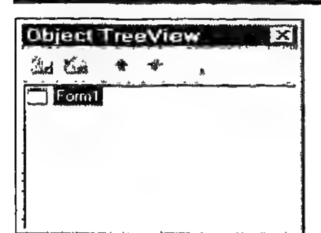
ونافذة لفحص الكائنات تسمى Object Inspector والموضحة بالشكل التالى :

شكل توضيحى :



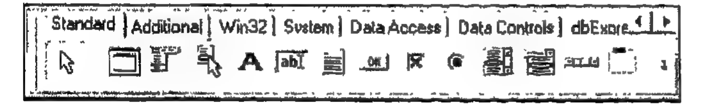
ونافذة للمشاهدة الشجرية للكائنات تسمى Object TreeView والموضحة بالشكل التالى :

شكل توضيحى :



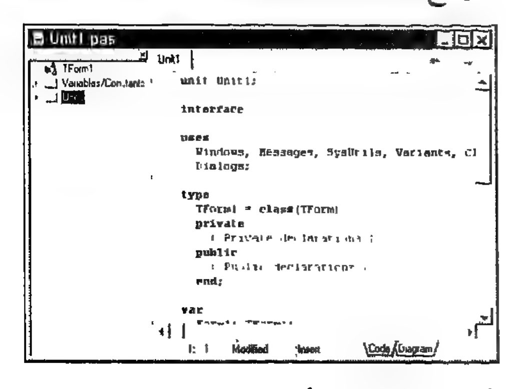
وباللية المكونات والموضحة بالشكل التالى :

شكل توضيحى :



ومحرر الكود البرمجى Code Editor الموضح بالشكل التالى :

شكل توضيحى :



ومدير المشروع Project Manager وأداة لإكتشاف الأخطاء البرمجية ومعالجتها تسمى debugger بالإضافة إلى العديد من الأدوات الأخرى.



بعض الأدوات قد لا تكون موجودة بكافة إصدارات Delphi 6.

وأنت تستطيع التنقل بحرية من التمثيل المرئى لأى كائن (موجود فى أداة تصميم الفورمة) إلى النافذة Object Inspector وذلك للتعديل فى الحالة التشغيلية الابتدائية لهذا الكائن ثم تنتقل بعد ذلك إلى محرر الكود البرمجى Code Editor للتعديل فى منطق التنفيذ لهذا الكائن.



التغـيير فى الخصائص المرتبطة بالكود البرمجى-مثل اسم أداة معاملة الحدث-فى الـنافذة Object Inspector يؤدى على الفور إلى تغيير الكود البرمجى المناظر لهذه الخصائص. بالإضـافة لذلـك فـإن الـتغيرات الـتى تجـرى عـلى الكود البرمجى-مثل تغيير اسلوب أداة معاملـة الحـدث فـى جملة الاعلان عن قطاع فورمة-تنعكس مباشرة داخل النافذة Object Inspector.

تعمـل بيـئة التطوير المتكاملة IDE على تدعيم إمكانية تطوير التطبيقات من خلال مـراحل الـدورة العمـرية للمنـتج-بدايـة مـن التصـميم إلى التطوير. كما أن استخدام الأدوات المتوفرة لدى بيئة التطوير المتكاملة IDE يسمح لك بأن تعد نموذج حقيقى وفعلى سريعا مما يؤدى بطبيعة الحال إلى التقليل من وقت الإعداد والتطوير.

تصميم التطبيقات

تمـتلك لغـة Delphi كافـة الأدوات الضرورية للبدء فى تصميم التطبيقات ومن بين هذه الأدوات ما يلى :

- نافذة فارغة تعرف بالفورمة form والتى يتم فيها تصميم واجهة الاستخدام User Interface (UI) للتطبيق الذى تعده.
- مجموعـة هائلـة مـن مكتـبات القطاعـات الـتى تشـتمل عـلى كمـية كبيرة جدا من الكائنات التى يمكن استخدامها أكثر من مرة.
- الـنافذة Object Inspector والتى يمكن من خلالها التحكم فى صفات وسلوك أى كائنات من خلال مجموعة من الخصائص والأحداث المرتبطة بهذا الكائن.
- محرر الكود البرمجى Code Editor الذى يمكن من خلاله التحكم فى منطق عمل التطبيق فى مرحلة التشغيل Run-Time.
- مديـر المشروع Project Manager لإدارة الملفات التى يتألف منها مشروع واحد أو عدة مشروعات.
- العديد من الأدوات الأخرى مثل محرر الصور الموجود بشريط الأدوات وأداة معالجة الأخطـاء المـتكاملة ID (Integrated Debugger) الموجـودة بـالقوائم لتدعيم عملية تطوير التطبيق فى بيئة التطوير المتكاملة IDE.

- أدوات الأوامر الخطية Command-Line والتى تتضمن المترجمات compilers وأدوات الربط Linkers بالإضافة إلى العديد من الخدمات الأخرى.

تستطيع أن تستخدم لغة Delphi لتصميم أى نوع من التطبيقات التى تعمل بنظم التشغيل الـ٣٢ بت–بداية من الخدمات العامة إلى البرامج المعقدة للوصول للبيانات والتعامل مع قواعد البيانات بالإضافة إلى التطبيقات الموزعة والتى تعمل بشبكات الحاسب.

أدوات قواعد البيانات وكذلك مكونات الارتباط بالبيانات التى تمتلكها لغة Delphi تسمح لك بأن تقوم سريعا بإعداد تطبيقات قواعد بيانات سواء كانت مكتبية Desktop (تعمل بأجهزة كمبيوتر مستقلة عن بعضها البعض) أو كانت شبكية Client/Server (التى تعمل بأجهزة كمبيوتر مرتبطة ببعضها البعض من خلال شبكة من الطراز Client/Server).

عن طريق استخدام أدوات التحكم الارتباط بالبيانات التى تمتلكها لغة Delphi تستطيع أن تشاهد على الطبيعة البيانات فى أثناء قيامك بتصميم التطبيق الخاص بك كما تشاهد على الفور نتائج عمليات الاستفسار التى تجرى بقواعد البيانات وكذلك التغييرات التى تجرى على واجهة استخدام التطبيق.

العديد من الكائنات المتوفرة فى مكتبة القطاعات يمكن الوصول إليها من خلال بيئة التطوير المتكاملة IDE وذلك من باللتية المكونات التى تقدم كافة أدوات التحكم سواء التى تكون ظاهرة فى مرحلة التشغيل Run-Time أو التى تكون مختفية فى مرحلة التشغيل Run-Time وكلاهما يمكن وضعه فى فورمة. ونود هنا القول بأن كل تبويب فى باللتية المكونات يشتمل على مجموعة من المكونات المتشابه فى الأداء الوظيفى. والعموم نجد أن اسماء الكائنات الموجودة فى مكتبات القطاع تبدأ بالحرف T مثل اسم الكائن TStatusBar.

إحدى الأشياء التى تعتبر من المظاهر القوية والفعالة التى تتمتع بها لغة Delphi تتمثل فى إنه يمكنك إنشاء المكونات الخاصة بك وذلك باستخدام object Pascal. وفى هذا الصدد نقول إن أغلب المكونات التى تأتى مع لغة Delphi قد تم كتابتها من خلال Object Pascal. وأنت تستطيع إضافة المكونات التى تكتبها بنفسك إلى باللتية المكونات كما تستطيع أيضا تفصيل باللتية تضع فيها المكونات التى تكتبها وذلك عن طريق ضم تبويبات جديدة إلى باللتية المكونات.

تستطيع أيضا استخدام لغة Delphi لإعداد تطبيقات تكون لديها القدرة على العمل بمختلف نظم التشغيل مثل الويندوز وLinux وذلك عن طريق استخدام CLX التى تشتمل على مجموعة من القطاعات التى لو تم استخدامها بدلا من القطاعات الموجودة فى VCL فإنها تسمح للبرنامج الذى تعده بأن يعمل بكل من بيئة الويندوز ونظام التشغيل Linux.

استخدام مكتبة المكونات

فى البداية نقول أن كل من VCL وCLX عبارة عن مكتبات قطاعات تتألف من مجموعة من الكائنات وبعض منها يتألف من مجموعة من المكونات أو أدوات التحكم والتى تستخدمها عند تطوير التطبيقات. وكلا المكتبتين تتشابهان معا إلى حد كبير وكلاهما يشتمل على العديد من الكائنات المتماثلة. وهناك بعض الكائنات الموجودة فى VCL تعمل على تنفيذ مظاهر وإمكانيات تكون متاحة ببيئة الويندوز فقط مثل الكائنات التى تظهر فى كل من التبويبات التالية بباليتة المكونات :

التبويب	أيقونات المكونات الموجودة به
ADO	Data Access | Data Controls | dbExpress | DataSnap | BDE | ADO | InterBa...
BDE	Data Access | Data Controls | dbExpress | DataSnap | BDE | ADO | InterBa...
QReport	InternetExpress | Internet | WebSnap | FastNet | Decision Cube | QReport | Di...
COM+	Decision Cube | QReport | Dialogs | Win 3.1 | Samples | ActiveX | COM+ | Ind...
Web Services	WebServices | InternetExpress | Internet | WebSnap | FastNet | Decision Cube...
Servers	Samples | ActiveX | COM+ | Indy Clients | Indy Servers | Indy Misc | Servers

أما بالنسبة للكائنات الموجودة فى CLX فهى متاحة بكل من بيئة الويندوز ونظام التشغيل Linux.

الكائنات الموجودة بكل من VCL وCLX تعتبر كيانات نشطة Active entities تشتمل على كافة البيانات الضرورية والأساليب (الكود) الذى يعمل على تعديل هذه البيانات التى تكون مخزنة فى حقول وخصائص الكائنات كما أن الكود يكون مؤلف من الأساليب التى تتعامل مع قيم الحقول والخصائص. هذا ويتم الاعلان عن كل كائن على أساس أنه قطاع Class. ونود هنا القول بأن كافة الكائنات الموجودة بكل من VCL وCLX تنحدر من الكائن TObject الذى يعتبر الجد الأصلى لهذه الكائنات بما فيها الكائنات التى تقوم بإعدادها بنفسك من خلال Object Pascal.

أى مجموعة فرعية من الكائنات يطلق عليها مكونات. ويمكن تعريف المكونات بأنها كائنات تستطيع وضعها فى فورمة أو Module بيانات والتعامل معها فى أثناء مرحلة التصميم Design-Time. هذا وتظهر المكونات فى بالليتة المكونات. وأنت تستطيع توصيف خصائص هذه المكونات بدون استخدام أى كود برمجى. ونود هنا القول بأن كافة المكونات بكل من VCL وCLX تنحدر من الكائن TComponent.

المكونات تعتبر كائنات لو نظرنا إليها من منظور أسلوب البرمجة OOP (اختصار للمصطلح Object-Oriented Programming) وذلك لأنها :

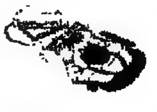
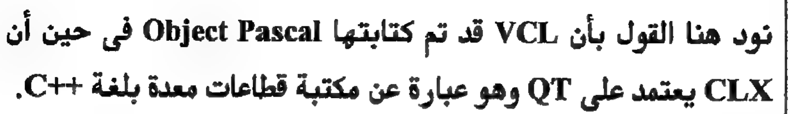
- تتمتع بمجموعة من الدوال التى يتم استخدامها للتعامل مع البيانات والوصول إليها بقواعد البيانات.
- تتوارث كل من البيانات وصفات السلوك من الكائنات التى تنحدر منها.
- تعمل بطريقة تفاعلية وتبادلية مع الكائنات التى تنحدر من نفس الجد وذلك فى إطار منظومة يطلق عليها polymorphism (تعدد الأشكال).

باختلاف معظم المكونات نجد أن الكائنات لا تظهر فى بالليتة المكونات. ولكن بدلا من ذلك يتم الاعلان عن متغير حالة افتراضية وهذا الاعلان يكون فى وحدة الكائن ويتم تلقائيا ولكن إذا لم يتم تلقائيا فإنه فى هذه الحالة ينبغى عليك القيام به بنفسك.

أدوات التحكم تعتبر نوعا خاصا من المكونات وتكون ظاهرة للمتسخدمين فى مرحلة التشغيل Run-Time. هذا وتعتبر أدوات التحكم مجموعة فرعية من المكونات. ويمكن القول بأن أدوات التحكم عبارة عن مكونات مرئية تستطيع أن تشاهدها عندما يكون التطبيق الذى تتولى إعداده فى حالة عمل. وكافة أدوات التحكم تمتلك عدد من الخصائص التى يمكن من خلالها تحديد فى الصفات الظاهرية لأدوات التحكم مثل الارتفاع Height والعرض Width. ونود هنا القول بأن كل من الخصائص والأساليب والأحداث التى تمتلكها كافة أدوات التحكم يتم توارثها من TControl.

الخصائص والأساليب والأحداث

كل من VCL وCLX تعمل على تكوين هياكل شجرية للكائنات المرتبطة ببيئة التطوير المتكاملة IDE الخاصة بلغة Delphi حيث تستطيع إعداد وتطوير التطبيقات بشكل سريع وفعال. والكائنات الموجودة بكلا المكتبتين تكون معتمدة على مجموعة من الخصائص والأساليب والأحداث. فكل كائن يتضمن عدد من الخصائص Properties التى تعرف بأنها أعضاء بيانات كما يمتلك عدد من الدوال التى تعمل على هذه البيانات وهذه الدوال يطلق عليه أساليب methods كما أن كل كائن لدية عدد من الطرق التى تتحكم فى تفاعله مع المستخدم وهذه الطريق تعرف بأنها أحداث Events.



نود هنا القول بأن VCL قد تم كتابتها Object Pascal فى حين أن CLX يعتمد على QT وهو عبارة عن مكتبة قطاعات معدة بلغة ++C.

الخصائص Properties

الخصائص Properties عبارة عن صفات أى كائن وهذه الصفات تؤثر بشكل مباشر سواء فى السلوك الظاهرى للكائن أو فى العمليات التى يجريها الكائن. فعلى سبيل المثال الخاصية Visible تعمل على تحديد ما إذا كان الكائن ظاهرا أم سيكون مختفيا بواجهة استخدام التطبيق فى أثناء تشغيل التطبيق. وفى هذا الصدد نقول إن الخصائص المصممة جيدا تجعل من السهولة بمكان للآخرين استخدام المكونات التى تضعها بالتطبيق كما تجعل أيضا من السهل عليك أن تجرى أى تعديلات ترغبها على هذه المكونات.

فيما يلى سنستعرض سويا المظاهر والإمكانيات المفيدة للخصائص :

- تختلف الخصائص عن الأساليب التى تكون متاحة فقط فى مرحلة التشغيل Run-Time فأنت تستطيع مشاهدة وتغيير قيم الخصائص فى مرحلة التصميم والحصول مباشرة على تغذية مرتدة كلما تغيرت الكائنات فى بيئة التطوير المتكاملة IDE.
- يمكن الوصول للخصائص من خلال النافذة Object Inspector حيث يمكنك إجراء أى تعديلات ترغبها على قيم خصائص الكائن الذى تتعامل معه. ونود هنا القول بأن تحديد قيم الخصائص فى مرحلة التصميم تكون أسهل من تحديدها عن طريق كود برمجى كما يجعل من الأسهل التعديل فى الأكواد البرمجية الموجودة بالتطبيق حيث ستكون الأكواد البرمجية أصغر حجما فى هذه الحالة.
- بسبب أن البيانات موجودة داخل الخصائص فإن ذلك يوفر لها نوع من الحماية والخصوصية بحيث تكون مرتبطة بالكائن ذاته.
- الاستدعاءات الحقيقية التى تتم سواء للحصول على قيم الخصائص أو لتعديلها يطلق عليها أساليب ومن ثم يكون فى الإمكان إجراء عملية معالجة خاصة للكائن بعيدا عن أعين المستخدمين للتطبيق فى مرحلة التشغيل Run-Time. فعلى سبيل المثال يمكن وضع بيانات فى جدول ولكن قد تبدو للمبرمج كما لو كانت عضو بيانات عادى.
- تستطيع أن تعد منهج منطقى يعمل على استثارة الأحداث أو تعديل بيانات أخرى فى أثناء الوصول للخاصية. فعلى سبيل المثال عملية تغيير قيمة خاصية واحدة قد تتطلب تعديل قيمة خاصية أخرى وهكذا...وأنت تستطيع إجراء التغيير فى قيمة الخاصية عن طريق الأساليب المعدة خصيصا لهذا الغرض.
- يمكن أن تكون الخصائص تصورية Virtual.
- هناك بعض الخصائص التى لا تكون مخصصة لكائن واحد فقط. فتغيير قيمة خاصية واحدة لأحد الكائنات يمكن أن يؤثر فى العديد من الكائنات الأخرى. فعلى سبيل المثال تحديد قيمة الخاصية Checked للكائن radio button يؤثر على الكائنات الآخرى التى من نفس النوع والموجودة معا فى مجموعة واحدة.



الأساليب Methods

الأسلوب method عبارة عن إجراء يكون دوما ملحقا بقطاع. والأساليب هى المسئولة عن تحديد سلوك أى كائن. وفى هذا الصدد نقول إن الأساليب الخاصة بأى قطاع يمكنها الوصول لكافة الخصائص سواء كانت عامة public أو محمية protected أو خاصة Private كما يمكنها الوصول أيضا لأعضاء البيانات للقطاع كما إنه من المعتاد أن يشار إليها على أساس كونها دوال member functions.

الأحداث Events

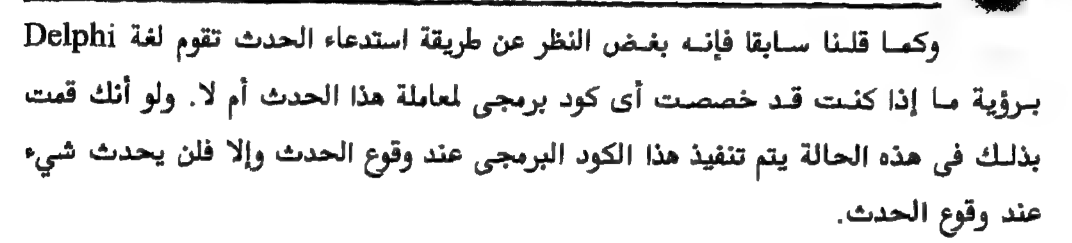
الحدث Event عبارة عن فعل أو واقعة يتحدد وقوعها من خلال البرنامج. وأغلب التطبيقات الحديثة يقال عنها أنها من الطائفة التى تعرف بـEvent-driven وذلك لأنه تم تصميم هذه التطبيقات بحيث تستجيب للأحداث التى تقع أثناء تفاعل المستخدمين معها. ففى أى برنامج لا يكون للمبرمج أى سبيل للتنبؤ بالتتابع المضبوط لوقوع الأحداث فى أثناء تعامل المستخدم مع البرنامج. فقد يقوم المستخدم باختيار عنصر من أحد القوائم الموجودة بالبرنامج أو قد يقوم بالنقر بالماوس على أحد المفاتيح أو الأيقونات أو قد يقوم بالتعليم على نص ما وهكذا...ولكن على العموم تستطيع كتابة كود برمجى لمعاملة الأحداث التى تربطها بالعناصر الموجودة بالبرنامج والتى تقع عند تعامل المستخدم مع هذه العناصر وذلك بدلا من أن تكتب كود برمجى دوما يتم تنفيذه بتتابع أو ترتيب ثابت.

بغض النظر عن الكيفية التى يتم بها استدعاء أى حدث فإن نركز على أن لغة Delphi تلقى نظرة لكى تشاهد لو أنك قمت بكتابة أى كود برمجى لمعاملة حدث ما أم لا. ولو كان ذلك حدث فإن يتم تنفيذ هذا الكود البرمجى ولكن خلاف ذلك تجد أنه يتم التعامل مع الحدث بالطريقة الطبيعية أو الإفتراضية.

أنواع الأحداث

أنواع الأحداث التى يمكن أن تحدث يمكن تصنيفها إلى صنفين أساسيين هما :

- أحداث المستخدم User Events.
- أحداث النظام System Events.



وكما قلنا سابقا فإنه بغض النظر عن طريقة استدعاء الحدث تقوم لغة Delphi برؤية ما إذا كنت قد خصصت أى كود برمجى لمعاملة هذا الحدث أم لا. ولو أنك قمت بذلك فى هذه الحالة يتم تنفيذ هذا الكود البرمجى عند وقوع الحدث وإلا فلن يحدث شيء عند وقوع الحدث.

أحداث المستخدم User Events

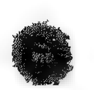
أحداث المستخدم User Events عبارة عن الأفعال التى تتم من خلال مستخدم البرنامج. والأمثلة على أحداث المستخدم User Events كثيرة ومتعددة منها الحدث OnClick الذى يقع عندما يقوم المستخدم بالنقر بالماوس على أى عنصر بالبرنامج وكذلك الحدث OnKeyPress والذى يقع عند يضغط المستخدم على أى مفتاح بلوحة المفاتيح بالإضافة إلى الحدث OnDblClick والذى يقع عندما ينقر المستخدم بالماوس نقرا مزدوجا. وهذه الأحداث غالبا ما تكون مرتبطة بالأفعال التى يفعلها مستخدم البرنامج.

أحداث النظام System Events

أحداث النظام System Events عبارة عن الأحداث التى تقع بواسطة نظام التشغيل نفسه. فعلى سبيل المثال الحدث OnTimer الذى يقع بواسطة المكون Timer حينما تنقضى فترة زمنية محددة مسبقا أما الحدث OnCreate فيقع عندما يتم إنشاء مكون فى حين أن الحدث OnPaint يقع عندما يكون هناك مكون أو نافذة فى حاجة لأن يتم إعاده رسمها وهكذا...ومن المعتاد أن لا تقع أحداث النظام System Events من خلال الأفعال التى يفعلها مستخدم البرنامج.

الـObject Pascal ومكتبات القطاعات

الـObject Pascal عبارة عن مجموعة من الامتدادات للغة Pascal القياسية التى تعتبر اللغة التى بنيت عليها لغة Delphi. وعن طريق استخدام كل من بالليتة المكونات والنافذة Object Inspector الخاصة بلغة Delphi تستطيع وضع مكونات VCL أو مكونات CLX بالفورمة وتتعامل مع خصائص كل منهم بدون كتابة أى كود برمجى.

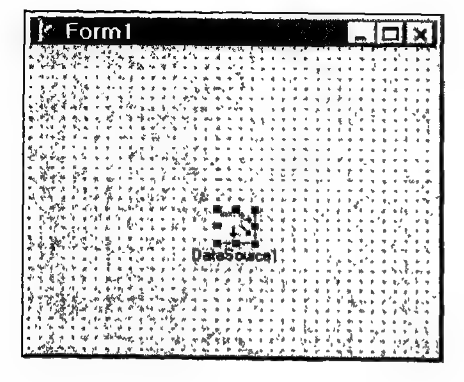


كافة الكائنات تنحدر من TObject وهو عبارة عن قطاع استخلاص يمتلك عدد من الأساليب تتحكم فى السلوك الوظيفى مثل البناء والهدم والتعامل مع الرسائل. كما أن TObject يصبح على الفور جد للعديد من القطاعات البسيطة.

المكونات الموجودة فى VCL أو CLX تنحدر من قطاع الاستخلاص TComponent. ونود هنا القول بأن المكونات عبارة عن كائنات تستطيع التعامل معها بالفورم فى مرحلة التصميم. هذا والمكونات المرئية-مثل المكونات TForm و TSpeedButton والتى تظهر على الشاشة فى مرحلة التشغيل Run-Time-يطلق عليها أدوات تحكم وهى تنحدر من TControl.

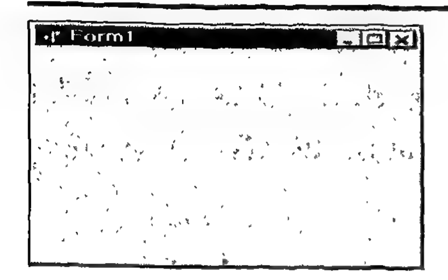
بالإضافة إلى مجموعة المكونات المرئية هناك مكتبات مكونات تشتمل على العديد من الكائنات الغير مرئية. ونود هنا القول بأن بيئة التطوير المتكاملة IDE تسمح لك بأن تضيف العديد من المكونات الغير مرئية إلى البرامج التى تعدها بنفسك وذلك عن طريق اسقاطهم فى الفورم. فعلى سبيل المثال لو إنك تكتب تطبيق يتصل بقاعدة بيانات فى هذه الحالة قد تحتاج لأن تضع المكون TDataSource بفورمة. هذا وبالرغم من أن المكون TDataSource غير مرئى إلا إنه يتم تمثيله بالفورمة بأيقونة كما هو موضح بالشكل التالى :

شكل توضيحى :



فى حين أن هذا المكون لا يكون ظاهرا بالفورمة فى مرحلة التشغيل Run-Time كما هو موضح بالشكل التالى :

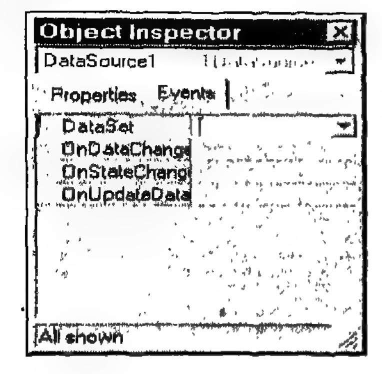
شكل توضيحي :



وأنـت تستطيع التعامل مع الخصائص والأحداث الخاصة بالمكون TDataSource فـي الـنافذة Object Inspector (كمـا هـو موضـح بالشـكلين التالـيين) كمـا هـو الحال مع المكونات المرئية :

شكل توضيحي :

شكل توضيحي :



عـندما تكتـب قطاعـات خاصـة بـك مـن خلال Object Pascal فينبغى أن تكون منحدرة من TObject فى مكتبة القطاع التى تخطط أن تستخدمها. هذا وعليك أن تستخدم VCL لو أنك تكتب تطبيق سيعمل ببيئة الويندوز فقط فى حين عليك أن تستخدم CLX لو كنـت تكتـب تطبـيقات تعمـل بكـل مـن بيـئة الويندوز ونظام التشغيل Linux. وعن طريق



إعداد قطاعات جديدة من قطاع التأسيس المناسب (أو واحد من القطاعات التى تنحدر منه) فإنك توفر القطاعات الخاصة بك لها أداء وظيفي محدد ولكن ينبغي عليك التأكد من أن هذه القطاعات تعمل بشكل جيد مع القطاعات الأخرى الموجودة فى مكتبة القطاع.

استخدام نموذج الكائن Object Module

أسلوب البرمجة OOP يعتبر الامتداد الطبيعى لأسلوب البرمجة الهيكلية الذى يركز فى المقام الأول على إعداد كود برمجي يمكن استخدامه أكثر من مرة كما يهدف أيضا إلى التكامل بين البيانات والأداء الوظيفي. وبمجرد أن تقوم بإنشاء كائن (أو قطاع حيث ذلك أكثر شيوعا الآن) فإنك تستطيع أنت والمبرمجين الآخرين استخدام هذا الكائن أو القطاع فى مختلف التطبيقات مما يؤدى إلى التقليل بقدر الإمكان من فترة إعداد التطبيقات وفى نفس الوقت يعمل على زيادة مستوى الإنتاجية لدى المبرمج.

ما هو الكائن Object

الكائن Object أو القطاع Class عبارة عن وعاء أو مخزن للبيانات والعمليات التى تجرى عليها فى وحدة واحدة. وفى هذا الصدد نقول أنه قبل إبتكار أسلوب البرمجة OOP كانت كل من البيانات والعمليات التى تجرى عليها (الدوال Functions) يتم التعامل معها على أساس أنها عناصر منفصلة عن بعضها البعض.

تستطيع أن تبدأ فى فهم الكائنات لو أنك فهمت سجلات كائن الباسكال أو الهياكل البنائية بلغة C. فالسجلات تتألف من حقول تشتمل على ألبيانات حيث أن كل حقل ينتمي لنوع معين. والسجلات تجعل من السهولة بمكان الإشارة إلى مجموعة من عناصر بيانات مختلفة فى النوع.

كذلك يمكن اعتبار الكائنات بمثابة مجموعات من عناصر البيانات. ولكن الكائنات–على خلاف السجلات–تكون مشتملة على عدد من الإجراءات والدوال التى تتعامل مع عناصر البيانات هذه. وهذه الإجراءات والدوال يطلق عليها أساليب Methods.

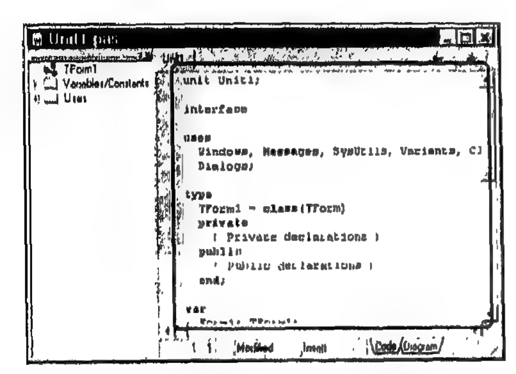
عناصر البيانات الموجودة بالكائن يمكن الوصول إليها عن طريق الخصائص الخاصة بالكائن. وخصائص الكائنات VCL وCLX لديها قيم يمكنك تغييرها فى مرحلة التصميم بدون كتابة كود برمجى للقيام بهذا التغيير. ولو أنك تحتاج لأن يتم تغيير قيمة خاصية فى مرحلة التشغيل Run-Time فى هذه الحالة ستحتاج لأن تكتب كود برمجى صغير الحجم للغاية.

الدمج بين البيانات والأداء الوظيفى فى وحدة واحدة يطلق عليه التغليف encapsulation أو التكامل . هذا وبالإضافة إلى مبدأ التغليف أو التكامل نجد أن أسلوب البرمجة OOP يتصف بصفة التوارث inheritance وبصفة التعددية polymorphism. ونود هنا القول بأن التوارث يعنى أن الكائنات تكتسب الأداء الوظيفى الخاص بها من كائنات أخرى (يطلق عليها الأسلاف ancestors) كذلك فإن الكائنات تستطيع تعديل سلوكها التى ورثته من الكائنات الأخرى. أما صفة التعددية فتعنى أن الكائنات المختلفة والتى تنحدر من نفس السلف تعمل على تدعيم نفس التفاعل مع الخصائص والأساليب والذى يطلق عليه فى الغالب التفاعلية التبادلية interchangeably.

اكتشاف الكائنات المتاحة لدى لغة Delphi

عندما تنشأ مشروع جديد تجد أن لغة Delphi تقدم لك فورمة جديدة لكى تقوم بتفصيلها. كما إنه فى محرر الكود البرمجى Code Editor تقوم لغة Delphi بالإعلان عن نوع قطاع جديد للفورمة وتعمل أيضا على إعداد الكود البرمجى الذى يتولى مهمة إنشاء حالة جديدة للفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :



الكود البرمجي الذى يتم تكوينه للتطبيق الجديد الذى يتم إعداده للعمل ببيئة الويندوز فقط يكون مشابه إلى حد كبير للكود البرمجي التالى :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls, Dialogs;
type
      TForm1 = class(TForm)      { The type declaration of the form
                                   begins here }
      private
            { Private declarations }
      public
            { Public declarations }
      end ;                      { The type declaration of the form ends here }
var
      Form1: TForm1;
implementation                   { Beginning of implementation part }
{$R *.DFM}
end.                             { End of implementation part and unit}
```



نوع القطاع الجديد عبارة عن TForm1 وهو منحدر من النوع TForm الذى يعتبر أيضا قطاع.

القطاع يشبه السجل على أساس أن كل منهما يشتمل على حقول بيانات ولكن القطاع يشتمل أيضا على أساليب–عبارة عن كود برمجي يؤثر فى البيانات المخزنة بالكائن. وحتى الآن يظهر TForm1 بحيث لا يشتمل على أى حقول أو أساليب وذلك لأنك لم تقم بعد بإضافة أى مكونات للفورمة (حقول الكائن الجديد) كما إنك لم تقم بإنشاء أى أدوات معاملة للأحداث (الأساليب الخاصة بالكائن الجديد). ولكن بالرغم من ذلك فإننا نقول إن TForm1 يشتمل على حقول وأساليب متوارثة حتى ولو لم تكن ترى أى منهم فى الكود البرمجي السابق.

فيما يلى جملة إعلان عن متغير يسمى Form1 ينتمى للنوع الجديد TForm1 :.

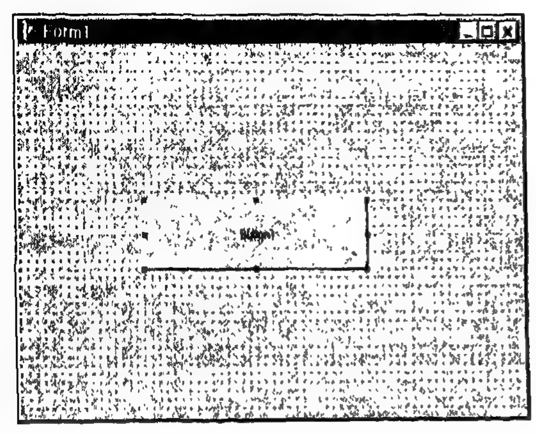
```
Var
      Form1: TForm1;
```

المتغير Form1 يمثل حالة instance أو كائن ينتمى لنوع القطاع TForm1. وأنت تستطيع الاعلان عن أكثر من حالة تنتمى لنوع قطاع وقد تجبر على القيام بذلك فى بعض الأحيان. فعلى سبيل المثال تحتاج لأن تعلن عن أكثر من حالة لكى تنشأ عدة نوافذ وليدة فى التطبيقات التى تشتمل على واجهة استخدام من النوع MDI (اختصار للمصطلح Multiple Document Interface). وفى هذا الصدد نقول إن كل حالة تحتفظ بالبيانات الخاصة بها ولكن فى نفس الوقت كافة الحالات تستخدم نفس الكود البرمجى لتنفيذ الأساليب الخاصة بها.

بالرغم من إنك لم تقم بعد بإضافة أى مكونات إلى الفورمة كما لم تقم بكتابة أى كود برمجى حتى الآن إلا إنك تمتلك الآن تطبيق يعتبر كامل وتستطيع ترجمتة Compile وتشغيله ولكن كل الذى سيظهر على الشاشة عند التشغيل عبارة عن فورمة فارغة فقط.

لتفترض أنك أضفت المكون Button إلى هذه الفورمة (بالنقر المزدوج بالماوس على أيقونة المكون Button بالليتة المكونات) لتصبح كما هو موضح بالشكل التالى :

شكل توضيحى :

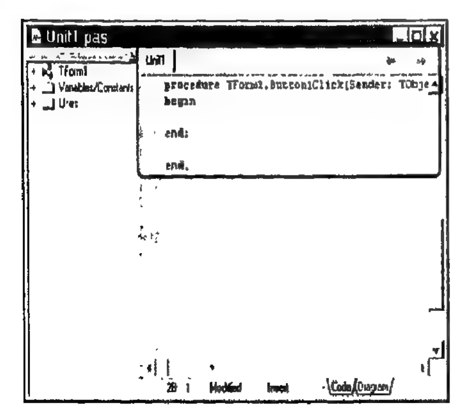


وترغب الآن فى كتابة الكود البرمجى لأداة معاملة الحدث OnClick الخاص بالمكون Button والذى يعمل على تغيير لون الفورمة لتصبح خضراء عندما يقوم المستخدم بالنقر بالماوس على المكون Button فى مرحلة التشغيل Run-Time.

وللقيام بذلك اتبع الخطوات التالية :

(١) انقر بالماوس نقرا مزدوجا على المكون Button بالفورمة ليظهر على الشاشة محرر الكود البرمجي Code Editor وبه أداة المعاملة للحدث OnClick الخاص بهذا المكون كما هو موضح بالشكل التالى :

شكل توضيحي :

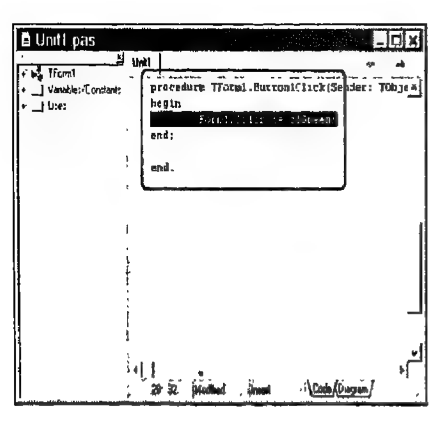


(٢) بالسطر الموجود به مؤشر الكتابة اكتب الكود التالى :

```
Form1.Color := clGreen;
```

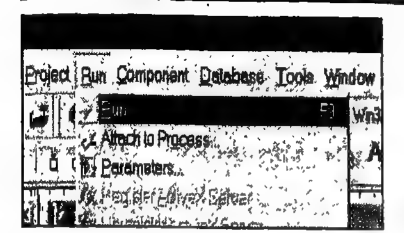
كما هو موضح بالشكل التالى :

شكل توضيحي :



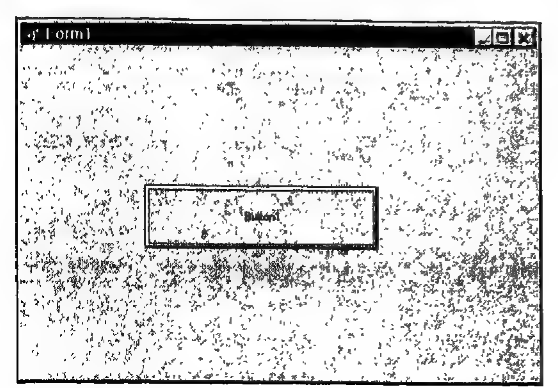
(٣) والآن قم بتشغيل البرنامج بالضغط على المفتاح F9 بلوحة المفاتيح أو انقر بالماوس على الأيقونة [أيقونة] بشريط الأدوات أو افتح القائمة Run واختر منها Run كما هو موضح بالشكل التالى :

شكل توضيحى :



(٤) لتظهر على الشاشة الفورمة وهى مشتملة على المفتاح كما هو موضح بالشكل التالى :

شكل توضيحى :



(٥) والآن انقر بالماوس على المفتاح Button1 لتجد أن الفورمة قد تغير لونها وأصبحت خضراء اللون كما هو موضح بالشكل التالى :

شكل توضيحى :



(٦) اغلق البرنامج بالنقر بالماوس على الزر (X) بالركن الأيمن العلوى للفورمة أو اضغط على المفتاحين Alt+F4 بلوحة المفاتيح أو عد مرة أخرى للغة Delphi وفيها اضغط على المفتاحين Ctrl+F2 بلوحة المفاتيح أو افتح القائمة Run واختر منها الأمر Program Reset كما هو موضح بالشكل التالى :



شكل توضيحى :

يمكـن للكائـنات أن تشـتمل عـلى كائـنات أخرى على أساس أنها حقول بيانات. ففـى كـل مـرة تقـوم فـيها بوضـع مكـون فـى فورمة يظهر على الفور حقل جديد فى جملة الإعـلان عـن الـنوع بمحـرر الكـود الـبرمجى Code Editor. ولو أنك قمت بإنشاء التطبيق السـالف الذكـر ثـم ألقيـت نظرة على الكود البرمجى الموجود الآن فى محرر الكود البرمجى Code Editor فسوف تشاهد الآتى :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls;
type
	TForm1 = class (TForm)
		Button1: TButton;	{ New data field }
		procedure Button1Click(Sender: TObject);	{ New
							method declaration }
	private
		{ Private declarations }
	public
		{ Public declarations }
	end ;
var
	Form1: TForm1;
implementation
{$R *.DFM}
procedure TForm1.Button1Click(Sender: TObject);	{ The code of
							the new method }
begin
	Form1.Color := clGreen;
end ;
end.
```

نود هنا القول بأن الكائن TForm1 يمتلك حقل يسمى Button1 وهو يناظر المفتاح الذى أضفته إلى الفورمة قبل ذلك. وفى هذا الصدد نقول إن TButton عبارة عن نوع قطاع ومن ثم فإن Button1 يشير إلى كائن.

أدوات معاملة كافة الأحداث التى تكتبها فى لغة Delphi تعتبر بمثابة أساليب لكائن الفورمة. ففى كل مرة تقوم فيها بإنشاء أداة معاملة لحدث فإنه يتم على الفور الإعلان عن أسلوب فى نوع كائن الفورمة. والآن أصبح النوع TForm1 مشتملا على أسلوب جديد وهو الإجراء Button1Click والذى تم الإعلان عنه داخل جملة الإعلان عن النوع TForm1. أما الكود البرمجى الذى يقوم بتشغيل الأسلوب Button1Click فيظهر فى الجزء implementation من الوحدة.

تغيير اسم مكون

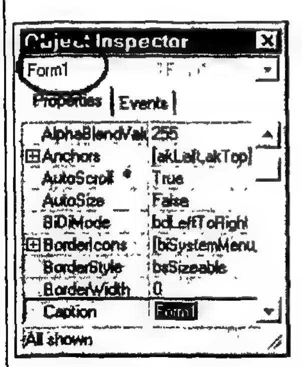
ينبغى عليك دوما استخدام النافذة Object Inspector لتغيير اسم أى مكون. فعلى سبيل المثال لنفترض أنك ترغب فى تغيير اسم فورمة من الاسم الافتراضى لها وهو Form1 ليصبح معبرا أكثر عن وظيفة الفورمة وليكن مثلا ColorBox. وللقيام بذلك اتبع الخطوات التالية :

(١) اجعل الفورمة هى العنصر المختار فى بيئة التطوير المتكاملة IDE كما هو موضح بالشكل التالى :

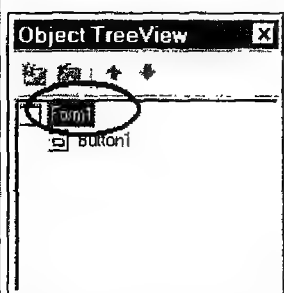
شكل توضيحى :



لكى تتأكد من أن الفورمة هى المختارة حاليا عليك أن تتأكد من إنك تشاهد Form1 بالقائمة المنسدلة الموجودة بقمة النافذة Object Inspector كما هو موضح بالشكل التالى :



كما تأكد أيضا من إن Form1 معلم عليها فى النافذة Object TreeView كما هو موضح بالشكل التالى :



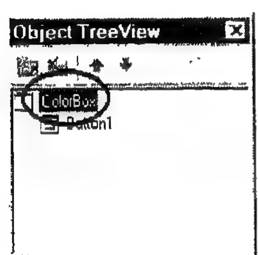
(٢) فى النافذة Object Inspector وعند الخاصية Name اكتب ColorBox ثم اضغط على المفتاح Enter بلوحة المفاتيح كما هو موضح بالشكل التالى :

شكل توضيحى :





تلاحظ أن اسم الفورمة أصبح **ColorBox** فى كل من النافذة **Object TreeView** كما هو موضح بالشكل التالى :



وبالقائمة المنسدلة الموجودة فى أعلى النافذة **Object Inspector** كما هو موضح بالشكل التالى :

من مظاهر القوة التى تتمتع بها لغة Delphi أنك عندما تقوم بتغيير قيمة الخاصية Name للفورمة من خلال النافذة Object Inspector تجد أن الاسم الجديد ينعكس مباشرة داخل الملف dfm. أو الملف xfm. الخاص بالفورمة (فى العادة لا يتم تعديل محتويات هذه الملفات يدويا) وكذلك فى الكود البرمجى الذى تكونه لغة Delphi وبالتالى سيصبح كالآتى :

```
unit Unit1;
interface
uses Windows, Classes, Graphics, Forms, Controls;
type
      TColorBox = class (TForm)  { Changed from TForm1 to TColorBox }
            Button1: TButton;
```



```
        procedure Button1Click(Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
    end ;
var
    ColorBox: TColorBox;        { Changed from Form1 to ColorBox }
implementation
{$R *.DFM}
procedure TColorBox.Button1Click(Sender: TObject);
begin
    Form1.Color := clGreen;     { The reference to Form1 didn't
                                change! }
end ;
end.
```

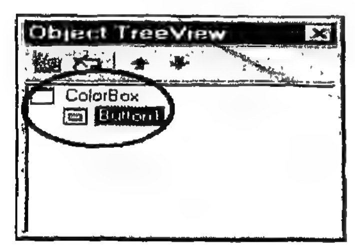
تلاحظ أن الكود البرمجي الموجود في أداة معاملة الحدث OnClick المرتبط بالمفتاح Button1 لم يحدث به أى تغيير. والسبب في ذلك أنك قمت بنفسك بكتابة هذا الكود البرمجي وليست اللغة ولذلك ينبغي عليك تحديثه بنفسك وتصحح أى إشارات مرجعية للفورمة وبالتالى يصبح كالآتى :

```
procedure TColorBox.Button1Click(Sender: TObject);
begin
  ColorBox.Color := clGreen;
end ;
```

توارث البيانات والكود البرمجي من كائن

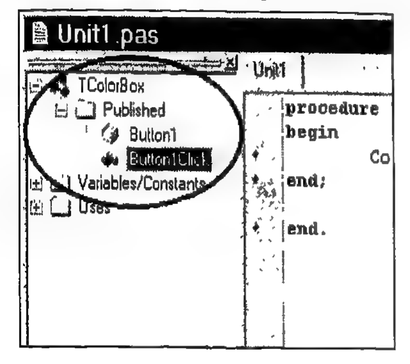
الكائن TForm1 الذى نتعامل معه حتى الآن يبدو أنه بسيطا. فهذا الكائن يظهر لكي يشتمل على حقل واحد وهو الحقل Button1 كما نرى في النافذة Object TreeView الموضحة في الشكل التالى :

شكل توضيحي :



كما إنه يشتمل أيضا على أسلوب واحد وهو الأسلوب Button1Click كما نشاهد في الجزء الأيسر بنافذة محرر الكود البرمجي Code Editor وكما هو موضح بالشكل التالـي :

شكل توضيحي :



فـي حين أنه لا يشتمل على أى خصائص. ومن ثم فأنت تستطيع إظهار أو إخفاء الفورمة أو تغيير حجمها كما تستطيع أيضا إضافة أو مسح أيقونات الإطار القياسى بالإضافة إلى إمكانـية تهيـئة الفورمة بحيث تصبح جزء من تطبيق MDI. وأنت تستطيع القيام بهذه الأشياء لأن الفورمة قد ورثت كافة الخصائص والأساليب الخاصة بالمكون TForm.

عـندما تضـيف فورمـة جديدة إلى المشروع الذى تعده فإنك تبدأ مع TForm وتعمل عـلى تفصـيله عـن طـريق إضافة مكونات له وتغيير قيم الخصائص الخاصة به وكتابة أكواد برمجية لأدوات معالجة الأحداث المرتبطة به.

عـلى العمـوم لكى تقوم بتفصيل أى كائن عليك أولا أن تسوق كائن جديد من كائن موجـود بالفعل فعندما تضيف فورمة جديدة إلى المشروع الذى تعده تقوم لغة Delphi تلقائيا بـDrive فورمة جديدة من النوع TForm وذلك من خلال الكود التالى :

```
TForm1 = class (TForm)
```

الكائن المساق يرث كافة الخصائص والأحداث والأساليب الخاصة بالكائن الأصلى.
ونـود هـنا القـول بـأن الكائن المساق يطلق عليه الكائن المنحدر descendant أو الخلف في حين أن الكائن الأصلى يطلق عليه السلف ancestor.



أى كائن يمكن أن يكون له سلف (جد) واحد فقط وفي نفس الوقت يمكن أن يكون له العديد من الأحفاد (الخلف).

فيما يلى سنستعرض سويا كل من الخصائص والأحداث والأساليب الخاصة بالكائن الجد TForm وذلك من خلال الجدول التالى :

الخصائص	الأساليب	الأحداث
Active	Cascade	OnActivate
ActiveControl	Next	OnClick
ActiveMDIChild	Previous	OnClose
BorderIcons	Tile	OnCloseQuery
BorderStyle	AfterConstruction	OnContextPopup
Canvas	BeforeDestruction	OnCreate
ClientHandle	Close	OnDblClick
ClientHeight	CloseQuery	OnDeactivate
ClientRect	Create	OnDestroy
ClientWidth	CreateNew	OnDragDrop
DesignerHook	DefocusControl	OnDragOver
DropTarget	Destroy	OnHelp
FocusedControl	FocusControl	OnHide
FormState	Hide	OnKeyDown
FormStyle	InvokeHelp	OnKeyPress
Icon	IsShortCut	OnKeyString
KeyPreview	Release	OnKeyUp
MDIChildCount	SetFocus	OnLoaded
MDIChildren	SetFocusedControl	OnMouseDown
Menu	Show	OnMouseMove
ModalResult	ShowModal	OnMouseUp
Parent	DisableAutoRange	OnMouseWheel
PixelsPerInch	EnableAutoRange	OnMouseWheelDown
Position	ScrollInView	OnMouseWheelUp
Scaled	ContentsRect	OnPaint
Visible	Broadcast	OnResize
WindowState	CanFocus	OnShortCut
AutoScroll	ContainsControl	OnShow
ChildHandle	ControlAtPos	
HorzScrollBar	CreateParented	
VertScrollBar	CreateParentedControl	
Bitmap	DisableAlign	
Brush	EnableAlign	
ClientOrigin	FindChildControl	
ControlCount	FlipChildren	

	Focused	Controls
	GetTabOrderList	Handle
	HandleAllocated	ParentWidget
	HandleNeeded	Showing
	InsertControl	TabOrder
	Invalidate	TabStop
	InvalidateRect	Action
	Realign	Align
	RemoveControl	Anchors
	Repaint	BoundsRect
	ScaleBy	Caption
	ScrollBy	Color
	Update	Constraints
	UpdateControlState	ControlState
	BeginDrag	ControlStyle
	BringToFront	Cursor
	ClientToParent	DragMode
	ClientToScreen	Enabled
	DragDrop	Font
	Dragging	Height
	EndDrag	HelpContext
	InitiateAction	HelpFile
	ParentToClient	HelpKeyword
	Refresh	HelpType
	ScreenToClient	Hint
	SendToBack	Left
	SetBounds	Name
	DestroyComponents	ParentFont
	Destroying	PopupMenu
	ExecuteAction	ShowHint
	FindComponent	Top
	FreeNotification	Width
	GetNamePath	ComponentCount
	GetParentComponent	ComponentIndex
	HasParent	Components
	InsertComponent	ComponentState
	IsImplementorOf	ComponentStyle
	ReferenceInterface	DesignInfo
	RemoveComponent	Owner
	RemoveFreeNotification	Tag



	SafeCallException SetSubComponent UpdateAction Assign ClassInfo ClassName ClassNameIs ClassParent ClassType CleanupInstance DefaultHandler Dispatch FieldAddress Free FreeInstance GetInterface GetInterfaceEntry GetInterfaceTable InheritsFrom InitInstance InstanceSize MethodAddress MethodName NewInstance	

مدى ومجال الاستخدام Scope وأدوات التأهيل Qualifiers

مجال الاستخدام Scope يعمل على تحديد إمكانية الوصول للحقول والخصائص والأساليب الخاصة بأى كائن. وفى هذا الصدد نقول إن كافة العناصر التى يتم الإعلان عنها داخل أى كائن تكون متاحة للاستخدام لهذا الكائن وأسلافه. وبالرغم من الكود البرمجى المسئول عن تنفيذ أسلوب ما يظهر خارج منطقة الأعلان بالكائن إلا إن الأسلوب يظل داخل مدى ومجال الكائن وذلك بسبب أنه تم الإعلان عنه داخل منطقة الإعلان بالكائن.

عندما تكتب كود برمجى لتنفيذ أسلوب ما يشير إلى خصائص أو أساليب أو حقول الكائن الذى تم به الإعلان عن هذا الأسلوب فى هذه الحالة لن تحتاج لأن تسبق أدوات التعريف Identifiers هذه باسم الكائن. فعلى سبيل المثال لو أنك وضعت مفتاح فى فورمة جديدة فإنك تستطيع كتابة الكود البرمجى التالى لأداة معاملة الحدث OnClick المرتبط بهذا المفتاح :

```
procedure  TForm1.Button1Click(Sender: TObject);
begin
        Color := clFuchsia;
        Button1.Color := clLime;
end ;
```

يمكن القول بأن أول جملة تكافئ الآتى :

```
Form1.Color := clFuchsia
```

وأنت لا تحتاج لأن تؤهل الخاصية Color بـ Form1 وذلك لأن الأسلوب Button1Click يعتبر جزء من TForm1 كما أن أدوات التعريف موجودة فى كتلة الأسلوب ومن ثم تقع داخل مدى الحالة TForm1 حيث يتم استدعاء الأسلوب.

على العكس من ذلك نجد أن الجملة الثانية تشير إلى لون كائن المفتاح (وليس إلى لون الفورمة حيث تم الإعلان عن أداة معاملة الحدث) ومن ثم فإنها تتطلب نوع من التأهيل.

تقوم لغة Delphi بإنشاء ملف لكل فورمة يكون مشتملا على الكود البرمجى الخاص بالفورمة (يسمى ملف الوحدة). ولو أنك رغبت فى الوصول لواحد من المكونات الموجودة بأى فورمة من خلال ملف الوحدة الخاص بفورمة أخرى فى هذه الحالة ستحتاج لأن تؤهل أسماء المكونات كالآتى :

```
Form2.Edit1.Color := clLime;
```

وبنفس الطريقة تستطيع أن تصل للأساليب الخاصة بأى مكون من فورمة أخرى. فعلى سبيل المثال :

```
Form2.Edit1.Clear;
```

لكى تصل إلى المكونات الموجودة فى الفورمة Form2 من ملف الوحدة الخاص بالفورمة Form1 فإنه ينبغى عليك فى هذه الحالة إيضا إضافة الوحدة الخاصة بالفورمة Form2 إلى الجملة uses الموجود فى الوحدة الخاصة بالفورمة Form1.

مدى استخدام كائن يمتد إلى الكائنات المنحدرة من هذا الكائن. وأنت تستطيع على كل حال أن تعيد الإعلان عن حقل أو خاصية أة أسلوب داخل أى من الكائنات المنحدرة. ومثل هذه العمليات (إعادة الإعلان) إما أن تكون مختفية أو متخطية override للعنصر المتوارث.

الإعلانات الخاصة والمحمية والعامة والمنشورة

عندما تعلن عن حقل أة خاصية أو أسلوب فى هذه الحالة تكون العضو الجديد لدية صفة الظهور والتى تحدد من خلال واحد من الكلمات المحجوزة التالية :

- Private
- Protected
- Public
- Published

صفة الظهور Visibility لأى عضو تحدد إمكانية وصولة إلى الكائنات والوحدات الأخرى.

- العضو الخاص Private يمكن الوصول إليه فقط من داخل الوحدة التى تم فيها الإعلان عن هذا العضو. وغالبا ما يتم استخدام الأعضاء الخاصة داخل قطاع وذلك للتعامل مع أساليب وخصائص أخرى (عامة أو منشورة).
- العضو المحمى protected يمكن الوصول إليه من داخل الوحدة التى فيها الإعلان عن القطاع الذى ينتمى إليه العضو كما يمكن الوصول إليه أيضا من داخل أى قطاع منحدر من هذا القطاع بغض النظر عن الوحدة الخاصة بالقطاع المنحدر.
- العضو العام Public يمكن الوصول إليه من أى موضع يمكن الوصول منه إلى الكائن الذى ينتمى إليه هذا العضو-بمعنى أنه يمكن الوصول إليه من الوحدة التى تم فيها الإعلان عن القطاع ومن أى وحدة أخرى تستخدم هذه الوحدة.
- العضو المنشور Published لدية نفس صفة الظهور التى يتمتع بها العضو العام ولكن مترجم اللغة Compiler يقوم هنا بتكوين معلومات من النوع التشغيلى runtime للأعضاء المنشورة.



الخصائص المنشورة (وهى خصائص الأعضاء المنشورة) تظهر فى النافذة Object Inspector فى مرحلة التصميم.

استخدام متغيرات الكائن Object Variables

تستطيع أن تخصص متغير كائن ما إلى متغير كائن آخر وذلك فى حالة أن المتغيرات تنتمى لنفس النوع أو هناك توافق بين الأنواع التى تنتمى إليها. وإذا نظرنا للموضوع من الناحية العملية نقول أنه يمكنك تخصيص متغير كائن إلى متغير كائن آخر لو أن نوع المتغير الذى يتم التخصيص له عبارة عن جد (سلف) لنوع المتغير الذى يتم تخصيصه. فعلى سبيل المثال الكود البرمجى التالى عبارة عن إعلان عن نوع TDataForm (VCL فقط) وفى جزء الإعلان عن متغير يتم الإعلان عن المتغيرين AForm و DataForm :

```
type
        TDataForm =  class (TForm)
        Button1: TButton;
                Edit1: TEdit;
                DataGrid1: TDataGrid;
                Database1: TDatabase;
        private
                { Private declarations }
        public
                { Public declarations }
        end ;
var
        AForm: TForm;
        DataForm: TDataForm;
```

المتغير AForm يكون من النوع TForm فى حين أن المتغير DataForm يكون من النوع TDataForm. هذا وحيث إن TDataForm يعتبر خلف لـ TForm فإن جملة التخصيص التالية تعتبر صحيحة وقانونية :

```
AForm := DataForm;
```

لنفترض أنك تكتب أداة معاملة حدث للحدث OnClick لمفتاح ما بحيث عند النقر بالماوس على هذا المفتاح فإنه يتم على الفور استدعاء أداة معاملة الحدث OnClick المرتبط بهذا المفتاح. وفى هذا الصدد نقول إن كل أداة معاملة من أدوات معاملة الأحداث



تكون لديها معامل يسمى Sender الذى ينتمى للنوع TObject كما هو موضح من خلال الكود البرمجى التالى :

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  ...
  ...
  ...
end ;
```

وبسبب كون المعامل Sender ينتمى إلى النوع TObject لذلك يمكن تخصيص أى كائن إلى هذا المعامل. ودوما تكون قيمة المعامل Sender عبارة عن أداة التحكم أو المكون الذى يستجيب للحدث. وأنت تستطيع اختبار المعامل Sender لكى تعثر على نوع المكون أو أداة التحكم التى تستدعى أداة معاملة الحدث وهذا الاختبار يتم عن طريق الكلمة المحجوزة is كما هو موضح من خلال الكود البرمجى التالى :

```
if Sender is TEdit then
        DoSomething
else
   DoSomethingElse;
```

إنشاء وتدمير الكائنات

العديد من الكائنات التى تستخدمها من خلال لغة Delphi مثل المفاتيح والحقول Edit Boxes تكون ظاهرة على الشاشة سواء فى مرحلة التصميم أو فى مرحلة التشغيل Run-Time. ولكن بعض الكائنات مثل صناديق الحوار الشائعة Common dialog boxes تظهر فى مرحلة التشغيل Run-Time فقط. كما أن هناك بعض الكائنات الأخرى مثل أدوات التوقيت Timers ومكونات مصادر البيانات تكون مختفية فى مرحلة التشغيل Run-Time.

فى كثير من الأحيان ترغب فى إنشاء كائنات خاصة بك. فعلى سبيل المثال قد تقوم بإنشاء كائن TEmployee يكون مشتملا على مجموعة من الخصائص مثل الاسم Name والعنوان Title والأجر فى الساعة HourlyPayRate. وأنت تستطيع بعد ذلك

إضافة الأسلوب CalculatePay والذى يستخدم البيانات الموجودة فى الخاصية HourlyPayRate لحساب مقدار ما سيتم دفعة للموظف. وفى هذا الصدد نقول إن الإعلان عن النوع TEmployee سيكون كما هو موضح من خلال الكود البرمجى التالى :

```
type
      TEmployee = class (TObject)
      private
            FName: string ;
            FTitle: string ;
            FHourlyPayRate: Double;
      public
            property Name: string read FName write FName;
            property Title: string read FTitle write FTitle;
            property HourlyPayRate: Double read FHourlyPayRate
            write FHourlyPayRate;
            function CalculatePay: Double;
      end ;
```

بالإضافة إلى الحقول والخصائص والأساليب التى قمت بتعريفها فإن TEmployee يرث كافة الأساليب الخاصة بـTObject. وأنت تستطيع وضع الإعلان عن نوع مثل الموجود فى الكود البرمجى السالف الذكر سواء فى الجزء interface أو الجزء implementation بأى وحدة وبعد ذلك تنشأ حالات للقطاع الجديد عن طريق استدعاء الأسلوب Create والذى يرثه TEmployee من TObject. هذا والكود البرمجى التالى يوضح لنا كيفية القيام بذلك :

```
var
      Employee: TEmployee;
begin
      Employee := TEmployee.Create;
end ;
```

الأسلوب Create يطلق عليه أداة إنشاء constructor. وهو يحدد موضع بذاكرة الكمبيوتر لكائن الحالة الجديدة ويعطى إشارة مرجعية إلى الكائن.

المكونات الموجودة بأى فورمة يتم إنشاؤها وتدميرها تلقائيا بواسطة لغة Delphi. ولكن لو أنك تكتب كود برمجى خاص بك للإعلان مبدئيا عن الكائنات فى هذه الحالة ستكون مسئولا عن عن تدمير هذه الكائنات أيضا. وفى هذا الصدد نقول إن كل كائن يرث الأسلوب Destroy (الذى يطلق عليه أداة الهدم destructor) من TObject.

على كل حال لتدمير كائن فإنه ينبغى عليك استدعاء الأسلوب Free (الذى يتم توارثه أيضا من TObject) وذلك لأن الأسلوب Free يتأكد من خلو الكائن من أى بيانات قبل أن يتم استدعاء الأسلوب Destroy.

على سبيل المثال الكود البرمجى التالى:

```
Employee.Free
```

يعمل على تدمير الكائن Employee ويقوم بإخلاء الذاكرة منه.

المكونات والملكية Ownership

تمتلك لغة Delphi آلية أساسية built-in لإدارة الذاكرة. وهذه الآلية تسمح لمكون أن يفرض مسئوليته عن تفريغ مكون آخر. وفى هذه الحالة نقول أن المكون الأول يمتلك المكون الثانى. وفى هذا الصدد نقول إن الذاكرة المخصصة للمكون المملوك يتم إخلاؤها تلقائيا عندما يتم تفريغ الذاكرة المخصصة للمكون المالك.

مالك أى مكون-القيمة المخصصة للخاصية Owner له-يتم تحديده عن طريق معامل يتم تمريره إلى أداة البناء فى أثناء إنشاء المكون. هذا ومن الطبيعي أن تمتلك أى فورمة كافة المكونات الموجودة بها كما إن الفورمة نفسها مملوكة للتطبيق الذى يشتمل على الفورمة. ومن ثم عند إنهاء التعامل مع التطبيق فإنه يتم على الفور وبشكل تلقائى تفريغ الذاكرة المخصصة لكافة الفورم والمكونات الموجودة بهم.

مبدأ الملكية هذا يتم تطبيقة فقط على TComponent وكافة المكونات التى تنحدر منه. فعلى سبيل لو أنك تنشأ الكائن TStringList أو الكائن TCollection (حتى ولو كان ملحقا بفورمة) فى هذه الحالة ستكون مسئولا عن تفريغ الكائن.

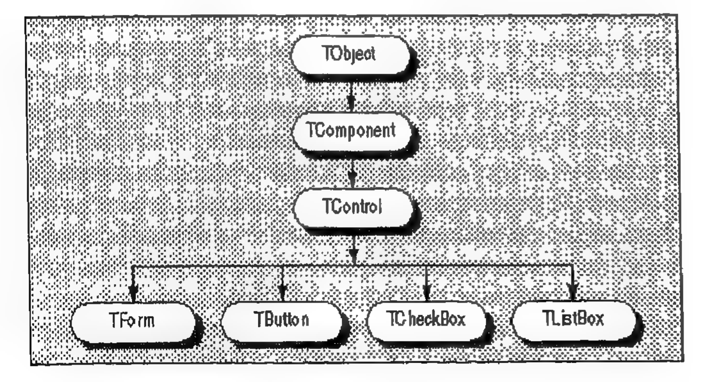


لا تخلط بين مالك المكون والمكون الأب لهذا المكون.

الكائنات والمكونات وأدوات التحكم

الديجرام الموضح في الشكل التالى يعتبر مشهد مبسط جدا للهيكل الشجرى للتوارث وهو يوضح العلاقة التبادلية بين الكائنات والمكونات وأدوات التحكم :

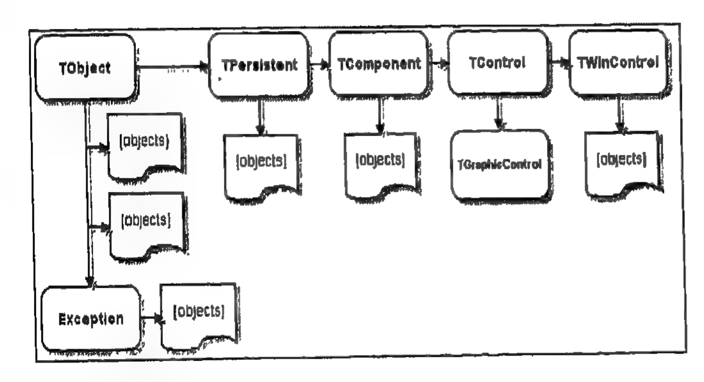
شكل توضيحي :



كل كائن يرث من TObject كما أن العديد من الكائنات ترث من TComponent. أما أدوات التحكم والتي ترث من TControl فلديها القدرة على عرض أنفسهم في مرحلة التشغيل Run-Time. فأداة تحكم مثل TcheckBox يرث كافة صفات الأداء الوظيفي الخاصة بكل من TObject وTComponent وTControl كما إنه يضيف قدرات وإمكانيات خاصة لنفسه.

الديجرام التالى عبارة عن مشهد عام لمكتبة المكونات المرئية VCL (اختصار للمصطلح Visual Component Library) والتي توضح الغالبية العظمى من التفرعات الخاصة بشجرة التوارث :

شكل توضيحي :







مكتبة المكونات CLX (اختصار للمصطلح Component Library for Cross-Platform) التى أعددتها شركة Borland تشبة إلى حد كبير الديجرام الموضح فى الشكل السابق ولكن الاختلاف الوحيد بينهما أن TWinControl تم استبدالها بـTWidgetControl.

بهـذا الديجـران نـرى العديد من قطاعات التأسيس الهامة والجدول التالى يقدم لنا وصفا لها :

القطاع	الوصف والاستخدام
TObject	يعتبر من أهم قطاعات التأسيس على الإطلاق وهو يعتبر الجد الأعلى لأى مكون فى المكتبة VCL أو المكتبة CLX. وهذا القطاع يعمل على توريث صفات السلوك الأساسية لكافة الكائنات سواء كانت VCL أو CLX وهذا التوريث يتم عن طريق مجموعة من الأساليب التى تؤدى الوظائف الأساسية مثل إنشاء وصيانة وتدمير أى حالة لأى كائن.
Exception	يعتبر قطاع التأسيس بكافة القطاعات التى ترتبط بالإستثناءات. والقطاع التأسيس هذا يعمل على توفير وسيط محكم ومتكامل لحالات الأخطاء كما يعمل أيضا على تمكين التطبيقات من التعامل مع حالات الخطأ بطريقة رائعة وفعالة.
TPersistent	قطاع التأسيس هذا مخصص لكافة الكائنات التى تنفذ الخصائص. هذا والقطاعات التى توجد تحت قطاع التأسيس TPersistent تتعامل مع إرسال البيانات إلى الـStreams وتسمح بتخصيص القطاعات.
TComponent	قطاع التأسيس هذا مخصص لكافة المكونات الغير مرئية مثل المكون TApplication. وهذا القطاع يعتبر الجد المشترك لكافة المكونات.

	وهو يسمح لأى مكون بأن يتم عرضه بباللية المكونات كما يسمح أيضا للمكون بأن يمتلك مكونات أخرى وكذلك يسمح للمكون بأن يتم التعامل معه مباشرة داخل أى فورمة.
TControl	عذا القطاع يمثل قطاع التأسيس لكافة أدوات التحكم التى تكون ظاهرة فى مرحلة التشغيل Run-Time. وهو يعتبر الجد المشترك لكافة المكونات المرئية ويعمل على توفير أدوات تحكم مرئية قياسية كثل Position وcursor كما يعمل أيضا على توفير أحداث تستجيب لأفعال الماوس.
TWinControl	قطاع التأسيس هذا مخصص لكافة الكائنات التى تستخدم فى تصميم واجهات الاستخدام UI وهو يطلق عليه أيضا Widgets. وفى هذا الصدد نقول إن أدوات التحكم الموجودة تحت قطاع التأسيس هذا يقال عليها أنها windowed controls ولديها القدرة على التقاط مدخلات لوحة المفاتيح. (فى مكتبة المكونات CLX نجد TWidgetControl بدلا من TWinControl).

فرع قطاع التأسيس TObject

فرع قطاع التأسيس TObject يشتمل على كافة الكائنات التى تنحدر من TObject وفى نفس الوقت لا يشتمل على الكائنات التى تنحدر من TPersistent. وفى هذا الصدد نقول إن كافة الكائنات الموجودة فى VCL أو فى CLX تنحدر من TObject عبارة عن قطاع استخلاص يمتلك عدد من الطرق التى تعمل على تعريف الصفات الأساسية للسلوك مثل الإنشاء والتدمير والرسائل أوتعامل النظام مع الأحداث. وأغلب القوة التى تتمتع بها الكائنات VCL وCLX يتم الحصول عليها من خلال الأساليب التى يقدمها TObject.

قطاع التأسيس TObject يعمل على توريث صفات السلوك الأساسية لكافة الكائنات فى VCL وCLX وذلك عن طريق توفير أساليب تعمل على توفير الآتى :



- القدرة على الاستجابة عندما يتم إنشاء حالات للكائن أو عندما يتم تدمير أى من هذه الحالات.
- النوع Class ومعلومات الحالة بأى كائن ومعلومات نوع مرحلة التشغيل Run-Time (RTTI) حول الخصائص المنشورة الخاصة به.
- الدعم الخاص بإمكانية التعامل مع الرسائل (بالنسبة لكائنات المكتبة VCL فقط).

قطاع التأسيس Tobject يعتبر الجد المباشر للعديد من القطاعات البسيطة. فالقطاعات التى تكون موجودة داخل هذا الفرع تشترك جميعها فى صفة واحدة وهامة جدا : فكلهم مؤقتون سريعوا الزوال. وهذا يعنى أن هذه القطاعات لا تمتلك أى أسلوب لحفظ الحالة التى يكونوا عليها قبل أن يتم تدميرهم ومن ثم يمكن القول بأنهم غير دائمين.

واحد من المجموعات الأساسية للقطاعات فى هذا الفرع هى القطاع Exception. فهذا القطاع يعمل على توفير مجموعة هائلة من قطاعات الاستثناء الأساسية Built-In والتى يتم من خلالها التعامل تلقائيا مع الأخطاء التى تنتج عن القسمة على أصفار وأخطاء التعامل مع ملفات I/O وأخطاء تفصيل أنواع خاصة غير صحيحة invalid typecasts بالإضافة إلى العديد من حالات الأخطاء الأخرى.

هناك نوع آخر من المجموعات فى الفرع TObject عبارة عن القطاعات التى تغلف هياكل البيانات مثل :

- القطاع TBits الذى يعمل على تخزين مصفوفة من القيم المنطقية Boolean.
- القطاع TList الذى يضم قائمة بالقطاعات المرتبطة معا.
- القطاع TStack الذى يضم مجموعة من المؤشرات الـlast-in first-out.
- القطاع TQueue الذى يضم مجموعة من المؤشرات الـfirst-in first-out.

فى المكتبة VCL تستطيع أيضا العثور على wrappers للكائنات الخارجية مثل TPrinter (الذى يغلف واجهة الاستخدام التى يتم من خلالها التعامل مع الطابعة المثبتة ببيئة الويندوز) وTRegistry الذى يعد Wrapper منخفضة المستوى لأحقية استخدام النظام والدوال التى تتعامل مع أحقية الاستخدام registry.



مثل هذه العناصر السالفة الذكر مخصصة للتطبيقات التى يتم تصميمها للعمل ببيئة الويندوز فقط.

الـTStream يعتبر مثال جيد لنوع آخر من القطاعات الموجودة فى هذا الفرع. وفى هذا الصدد نقول إن TStream عبارة عن نوع قطاع التأسيس لكائنات التدفق Stream Objects والتى يمكن الحصول منها على معلومات (أو تخصيص لها بيانات) من خلال أنواع مختلفة من وسائط التخزين مثل الاسطوانات التخزين (الصلبة أو المرنة) والذاكرة الديناميكية.

لقد رأيت أن هذا الفرع يضم العديد من القطاعات المتعددة الأنواع والتى تعتبر مهمة جدا بالنسبة لك فى أثناء إعداد وتطوير البرامج من خلال لغة Delphi.

فرع قطاع التأسيس TPersistent

الكائنات الموجودة فى هذا الفرع من المكتبة VCL والمكتبة CLX تنحدر من قطاع التأسيس TPersistent الذى يعمل على إضافة صفة الاستمرار persistence. وهذه الصفة تحدد ما الذى يتم حفظه مع ملف الفورمة أو Module البيانات وما الذى سيتم تحميله داخل الفورمة أو Module البيانات عندما يتم استراجع أى منهما من الذاكرة.

الكائنات الموجودة فى هذا الفرع تقدم عدد من الخصائص للمكونات. وهذه الخصائص هى الوحيدة التى يتم تحميلها وحفظها مع أى فورمة وذلك لو أن لديهم مالك. وفى هذا الصدد نقول إن المالك ينبغى أن يكون مكون. وهذا الفرع يقدم الدالة GetOwner التى تسمح لك بأن تحدد المالك لأى خاصية.

الكائنات الموجودة فى هذا الفرع تعتبر أولى الكائنات التى تشتمل على جزء النشر والذى يمكن فيه تحميل وحفظ الخصائص تلقائيا. كذلك فإن الأسلوب DefineProperties يسمح لك بأن تحدد كيفية تحميل وحفظ الخصائص.

فيما يلى سنستعرض سويا بعض من القطاعات الأخرى الموجودة فى الفرع TPersistent :



- القطاع TGraphicsObject وهو عبارة عن قطاع تأسيس مستخلص للكائنات الرسومية (التى تستخدم فى الرسم) مثل الكائن TBrush والكائن TFont والكائن TPen.
- القطاع TGraphic وهو عبارة عن قطاع تأسيس مستخلص لكائنات مثل الأيقونات والصور النقطية bitmaps والتى يمكنها تخزين وعرض صور مرئية : TBitmap و TIcon (وTMetafile المخصصة لبيئة الويندوز فقط).
- القطاع TStrings وهو عبارة عن قطاع تأسيس مستخلص للكائنات التى تمثل مجموعة من السلاسل الحرفية.
- القطاع TClipboard الذى يشتمل على النصوص أو العناصر الرسومية التى تم قصها أو نسخها من أى تطبيق.
- كل من القطاع TCollection والقطاع TOwnedCollection والقطاع TCollectionItem. وهذه القطاعات تستخدم للاحتفاظ بمجموعات مفهرسة من العناصر الخاصة.

فرع قطاع التأسيس TComponent :

فرع قطاع التأسيس TComponent يشتمل على الكائنات التى تنحدر من TComponent وفى نفس الوقت لا تنحدر من TControl. والكائنات الموجودة فى هذا الفرع عبارة عن مكونات تستطيع التعامل معها بالفورم فى مرحلة التصميم. وهى تعتبر كائنات لها صفة الاستمرار ولديها القدرة على القيام بالآتى :

- الظهور فى بالليتة المكونات مع إمكانية تغييرها فى أداة تصميم الفورمة.
- امتلاك وإدارة مكونات أخرى.
- تحميل وحفظ أنفسها.

هناك العديد من الأساليب فى TComponent تعمل على تحديد طريقة عمل المكونات فى مرحلة التصميم كما تحدد أيضا المعلومات التى سيتم حفظها مع كل مكون. ونود هنا القول بأن مبدأ الـStreaming تم التقديم له فى هذا الفرع بكل من المكتبة VCL

والمكتبة CLX. وفي هذا الصدد نقول إن لغة Delphi تقوم بتنفيذ أغلب العمليات التي تعتمد على مبدأ الـStreaming. أما بالنسبة للخصائص فيكون لها صفة الوجود والاستمرار في حالة أنه قد تم نشرهم ونود هنا القول بأن الخصائص المنشورة تكون Streamed تلقائيا.

بالإضافة لما سبق نجد أن قطاع التأسيس TComponent يعمل أيضا على تقديم المفهوم الأساسي لمبدأ الملكية Ownership الشائع بين الكائنات الموجودة بكل من المكتبة VCL والمكتبة CLX. وفي هذا الصدد نقول إن هناك خاصيتين تعملان على تدعيم مبدأ الملكية وهما الخاصية Owner والخاصية Components. هذا وكل مكون لدية الخاصية Owner التي تشير إلى مكون آخر على اساس أنه المالك له. فالمكون يمكن أن يمتلك مكونات أخرى. وفي هذه الحالة يتم الإشارة مرجعيا إلى كافة المكونات المملوكة في الخاصية Array للمكون المالك.

أداة البناء الخاصة بالمكون تتعامل مع معامل واحد فقط يتم استخدامه لتحديد المالك الجديد للمكون. ولو أن المالك موجود بالفعل في هذه الحالة يتم إضافة مكون جديد إلى قائمة المكونات المملوكة للمالك. هذا وبعيدا عن استخدام قائمة المكونات للإشارة مرجعيا إلى المكونات المملوكة فإن هذه الخاصية يمكن استخدامها أيضا لعمل تدمير تلقائي للمكونات المملوكة. وطالما أن المكون لديه مكون فإنه في هذه الحالة يتم تدمير هذا المكون عندما يتم تدمير المالك له. فعلى سبيل المثال حيث إن TForm ينحدر من TComponent لذلك فإن كافة المكونات المملوكة للفورمة يتم تدميرها وإخلاء مواقعها بالذاكرة عندما يتم تدمير الفورمة. وهذا المبدأ يفترض أن كافة المكونات الموجودة بالفورمة تعمل على تنظيف نفسها بطريقة جيدة عندما يتم استدعاء أدوات الهدم الخاصة بهم.

لو أن نوع الخاصية كان TComponent أو أحد المكونات التي تنحدر منه في هذه الحالة يقوم نظام الـStreaming بإنشاء حالة لهذا النوع عندما يتم قراءته. أما لو كان نوع الخاصية عبارة عن TPersistent وفي نفس الوقت ليس TComponent فإن نظام الـ Streaming يستخدم الحالة الموجودة حاليا والمتاحة من خلال الخاصية ويقوم بقراءة القيم الخاصة بخصائص هذه الحالة.



عند إنشاء ملف فورمة (ملف يتم استخدامه لتخزين معلومات عن المكونات الموجودة فى الفورمة) فإن أداة تصميم الفورمة تدور عبر مصفوفة المكونات الخاصة به ويحفظ كافة المكونات الموجودة فى الفورمة. وكل مكون يكون لدية علم عن كيفية كتابة خصائصة المتغيره إلى Stream (فى هذه الحالة يكون الـStream عبارة عن ملف نصى). وعلى النقيض عند تحميل خصائص المكونات فى ملف الفورمة فإن أداة تصميم الفورمة يدور داخل مصفوفة المكونات ويقوم بتحميل كل مكون على حدة.

أنواع تاقطاعات التى ستجدها فى هذا الفرع تتضمن الآتى :

- القطاع TMainMenu وهو يعمل على توفير شريط قائمة والقوائم المنسدلة منه drop-down وذلك داخل أى فورمة.
- القطاع TTimer وهو يتضمن دوال التوقيت.
- كل من القطاع TOpenDialog والقطاع TSaveDialog والقطاع TFontDialog والقطاع TColorDialog وغيهم من القطاعات الأخرى تعمل على توفير صناديق الحوار الشائعة الاستخدام.
- القطاع TActionList الذى يحتفظ بقائمة تتضمن الأفعال المستخدمة مع المكونات وأدوات التحكم مثل العناصر الموجودة بالقوائم والمفاتيح.
- القطاع TScreen والذى يتتبع الفورم وModules البيانات التى تم إعداد حالات إبتدائية لها بواسطة التطبيق كما يحدد الفورمة النشطة حاليا كما يعمل أيضا على نقل دفة التحكم داخل الفورمة النشطة حاليا كما يتتبع أيضا كل من حجم ودرجة وضوح الشاشة والمؤشرات (سواء الكتابة أو الماوس) بالإضافة إلى الفونتات المتاحة للتطبيق لكى يستخدمها.

المكونات التى لا تحتاج لواجهة استخدام مرئية يمكن الحصول عليها مباشرة من TComponent. هذا ولكى تصنع أداة مثل معدة التوقيت TTimer فإنك تستطيع فى هذه الحالة أن تنشأ هذه الأداة مباشرة من TComponent. وهذا النوع من المكونات يوجد فى بالليتة المكونات ولكنه يؤدى وظائف داخلية يمكن التحكم فيها من خلال الكود البرمجى وذلك بدلا من الظهور فى واجهة الاستخدام فى مرحلة التشغيل Run-Time.



في المكتبة CLX نجد أن الفرع TComponent يشتمل أيضا على THandleComponent. وهو عبارة عن قطاع تأسيس خاص بالمكونات الغير مرئية التي تتطلب نوع من المعاملة في أثناء مرحلة التشغيل Run-Time مثل صناديق الحوار والقوائم.

فرع قطاع التأسيس TControl

فرع قطاع التأسيس TControl يتألف من المكونات التي تنحدر من TControl وليس من TWinControl (قطاع التأسيس TWidgetControl في المكتبة CLX). والكائنات الموجودة في هذا الفرع عبارة عن أدوات تحكم هي في حد ذاتها كائنات مرئية بمعنى أن مستخدم التطبيق يستطيع رؤيتها والتعامل معها في مرحلة التشغيل Run-Time.

هذا وكافة أدوات التحكم لديها خصائص وأساليب وأحداث تتحكم في مظهر أدوات التحكم على الشاشة مثل موضعه والمؤشر المرتبط بنافذة أداة التحكم (أو الـWidget في المكتبة CLX) وطرق رسم أو نقل أداة التحكم والأحداث التي تقع عندما تستجيب أداة التحكم لأفعال الماوس.



أدوات التحكم لا تستقبل أي مدخلات من لوحة المفاتيح.

في حين أن TComponent يعمل على تعريف وتحديد سلوك كافة المكونات التي تتبعه نجد أن TControl يعمل على تعريف وتحديد سلوك كافة أدوات التحكم المرئية فقط. وهذا يتضمن روتينات الرسم والأحداث القياسية والملكية.

هناك نوعين أساسيين من أدوات التحكم هما :

- أدوات تحكم لديها نافذة خاصة بها.
- أدوات تحكم تستخدم نافذة أدوات التحكم التي تكثل الأب لها.

أدوات التحكم التى لديها نافذة خاصة بها يطلق عليها Windowed Controls (فى المكتبة VCL) أو يطلق عليها Widget-based Controls (فى المكتبة CLX) وهى تنحدر من TWinControl (بالمكتبة VCL) أو من TWidgetControl (بالمكتبة CLX).



كل من المفاتيح ومربعات الاختبار تنتمى لهذه النوعية من أدوات التحكم.

أما أدوات التحكم التى تستخدم النافذة الخاصة بأداة التحكم الأب فيطلق عليها أدوات تحكم رسومية Graphic وهى تنحدر من TGraphicControl.



كل من أداة التحكم Image وأداة التحكم Label تنتمى إلى هذه النوعية من أدوات التحكم.

فى المكتبة VCL نجد أن الاختلاف الأساسى بين هذه الأنواع من المكونات يتمثل فى أن أدوات التحكم الرسومية لا تمتلك نافذة معاملة ومن ثم فهى لا تستطيع استقبال دفة التحكم لإدخال البيانات.

أما فى المكتبة CLX نجد أن الإختلاف الأساسى بين هذه الأنواع من المكونات يتمثل فى أن أدوات التحكم الرسومية لا تمتلك Widget خاصة بها ومن ثم لا تتمكن من استقبال دفة التحكم لإدخال البيانات كما إنها لا تستطيع أيضا احتواء أدوات تحكم أخرى داخلها.

بسبب أن أدوات التحكم الرسومية لا تحتاج لمعاملة لذلك فإن متطلبتها من مصادر نظام التشغيل تكون فى أدنى مستوى كما أن رسم أى أداة من أدوات التحكم الرسومية يكون أسرع من رسم نفس النوع من أدوات التحكم بالمكتبة CLX.

أدوات التحكم التى تنحدر من TGraphicControl يجب أن ترسم نفسها بالفورمة وهى تتضمن أدوات التحكم التالية :

أداة التحكم	الأيقونة	الصفحة فى باليتة المكونات	الوصف والاستخدام
TImage		Additional	عرض الصور الرسومية.
TLabel		Standard	عرض نص بالفورمة.
TBevel		Additional	تمثيل التخطيط البارز أو المحفور.
TPaintBox		System	توفير نسيج يمكن للتطبيقات استخدامه لرسم صور أو إعادة تكوينها Rendering.



لاحظ أن أدوات التحكم هذه تضم نفس روتينات الرسم (مثل Repaint وInvalidate وغيرها)ومن ثم فهى لا تحتاج أبدا لاستقبال دفة التحكم فى أثناء مرحلة التشغيل Run-Time.

فرع قطاع التأسيس TWinControl

فرع قطاع التأسيس TWinControl (وقطاع التأسيس TWidgetControl فى المكتبة CLX) يتضمن كافة أدوات التحكم التى تنحدر من TWinControl الذى يعتبر قطاع التأسيس لكافة أدوات التحكم التى لها نافذة بما فيها العديد من العناصر التى ستسخدمها فى أثناء إعداد واجهة الاستخدام للتطبيق الذى تعده.

أما بالنسبة لـــTWidgetControl فهو يعتبر قطاع التأسيس لكافة أدوات التحكم التى لها نافذة والموجودة بالمكتبة CLX. وهى تستخدم أيضا فى إعداد واجهات الاستخدام للتطبيقات. والأمثلة على هذه النوعية من أدوات التحكم كثيرة ومنها المفاتيح والعناوين وشرائط الحركة Scroll bars سواء الأفقية أو الرأسية.

فيما يلى المظاهر والإمكانيات التى تتمتع بها أدوات التحكم التى لها نوافذ سواء التى توجد فى المكتبة VCL أو فى المكتبة CLX :



- هـذه النوعـية مـن أدوات التحكم يمكنها استقبال دفة التحكم فى أثناء التعامل مع التطبيق فى مرحلة التشغيل Run-Time.
- أدوات الـتحكم الأخـرى قـد تعـرض بيانات ولكن المستخدم لا يتمكن من استخدام لوحة المفاتيح للتفاعل مع هذه النوعية من أدوات التحكم.
- يمكن لهذه النوعية من أدوات التحكم اشتمال أدوات تحكم أخرى داخلها.
- أداة الـتحكم الـتى تشـتمل عـلى أدوات تحكم أخرى يطلق عليها الأب Parent أو الأصل وفى هذا الصدد نقول أن هذه النوعية من أدوات التحكم يمكن أن تكون أب لأداة تحكم واحدة أو أكثر (يطلق عليها أدوات التحكم الوليدة Child أو التابعة.
- هذه النوعية من أدوات التحكم لديها ما يعرف نافذة المعاملة window handle.

الكائـنات الـتى تـنحدر مـن TWinControl (أو مـن TWidgetControl بالمكتبة CLX) عـبارة عـن أدوات تحكـم تسـتطيع اسـتقبال دفة التحكم مما يعنى أن لديها القدرة عـلى اسـتقبال المدخـلات من مستخدم التطبيق من خلال لوحة المفاتيح. وهذا يؤكد على أن أدوات التحكم هذه لديها عدد أكبر من الأحداث.

هـذا الفـرع يتضـمن كـل من أدوات التحكم التى يتم رسمها تلقائيا (مثل TEdit و TListBox وTComboBox وTPageControl وغيرهـم كـثير...) وأدوات التحكم المفصلة الـتى يجـب على لغة Delphi رسمها (مثل TDBNavigator وTMediaPlayer (بالمكتبة VCL فقط) وTGuage (بالمكتبة VCL فقط) وغيرهم كثير...).

الكائـنات الـتى تـنحدر مباشـرة مـن TWinControl (أو مـن TWidgetControl بالمكتـبة CLX) تعتـبر أدوات تحكـم قياسية مثل الحقول والقوائم المنسدلة وقوائم العرض و Page Control ومن ثم لديها القدرة على رسم نفسها بالفورمة.

القطاع TCustomControl يتم توفيره من أجل المكونات التى تتطلب نافذة معاملة ولكـن لا تشـتمل عـلى أداة من أدوات التحكم القياسية والتى تتضمن القدرة على إعادة رسم نفسـها بنفسـها. هـذا ويـنبغى علـيك ألا تقلـق أبدا بخصوص الطريقة التى تقوم بها أدوات الـتحكم لإعـادة تكوين نفسها أو بخصوص طريقة استجابتهم للأحداث التى تقع لهم حيث إن لغة Delphi هى التى تتولى بالكامل تنفيذ مثل هذه العمليات بدلا منك.

الخصائص المشتركة لأدوات التحكم المنحدرة من قطاع التأسيس TControl

كافة أدوات التحكم المرئية (التى تنحدر من قطاع التأسيس TControl) تتشارك معا فى خصائص معينة من بينها ما يلى :

- خصائص الأفعال Action Properties.
- الخصائص التى تحدد الموضع والحجم والضبط Alignment.
- الخصائص التى تحدد طريقة العرض.
- خصائص الأب Parent Properties.
- خاصية التجول Navigation Property.
- خصائص السحب والإسقاط Drag-and-Drop.
- خصائص السحب والإرساء Drag-and-Dock (للكائنات الموجودة بالمكتبة VCL فقط).

فى حين أن هذه الخصائص يتم توارثها من قطاع التأسيس TControl إلا إنه يمكن نشرها—من ثم تظهر فى النافذة Object Inspector—فقط للمكونات التى يمكن تطبيق هذه الخصائص عليها. فعلى سبيل المثال نقول أن TImage لا يعمل على نشر الخاصية Color الخاصة به ومن ثم فإنه يتم تحديد لونه عن طريق الرسمة التى يتم عرضها داخله.

خصائص الأفعال Action Properties

الأفعال تسمح لك بأن تستخدم نفس الكود البرمجى لأداء العديد من الأفعال (فعلى سبيل المثال عندما يقوم كل من أيقونة فى شريط أدوات وعنصر بقائمة بعمل نفس الشئ) بالإضافة إلى توفير طريقة واحدة ومركزية لتنفيذ الأفعال أو تعطيلها وذلك بناء على الحالة التى يكون عليها التطبيق فى مرحلة التشغيل Run-Time.

الخاصية	الوصف والاستخدام
Action	تمييز وتحديد الفعل المرتبط بأداة التحكم.
ActionLink	هذه الخاصية تشتمل على كائن ربط الفعل الملحق بأداة التحكم.

خصائص تحديد الموضع والحجم والضبط Alignment

هـذه المجموعة من الخصائص تعمل على تحديد كل من موضع وحجم أداة التحكم داخل أداة التحكم الأصلية أو الأب :

الخاصية	الوصف والاستخدام
Height	تحديد ارتفاع (الحجم الرأسى) أداة التحكم.
Width	تحديد عرض (الحجم الأفقى) أداة التحكم.
Top	تحديد موضع الحافة العليا لأداة التحكم.
Left	تحديد موضع الحافة اليسرى لأداة التحكم.
AutoSize	تقرير ما إذا كانت أداة التحكم تقوم بتغيير حجمها تلقائيا بحيث يصبح حجمها مناسب لمحتوايتها.
Align	تحديد الجانب الذى ستوجد به أداة التحكم داخل أداة التحكم الـ Parent التى توجد بها (لليمن أو لليسار أو فى الوسط).
Anchor	تحديد طريقة تعليق أداة التحكم داخل أداة التحكم الـParent التى توجد بها.



الخاصية Anchor تكون للكائنات الموجودة فى المكتبة VCL فقط.

المجموعـة التالـية من الخصائص تعمل على تحديد كل من ارتفاع وعرض والحجم الكلى ل منطقة الـClient الخاصة بأداة التحكم:

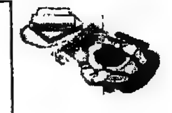
الخاصية	الوصف والاستخدام
ClientHeight	تحديـد ارتفـاع منطقة الـClient الخاصة بأداة التحكم وهذا الارتفاع يقاس بالبكسل.
ClientWidth	تحديـد عـرض منطقة الـClient الخاصة بأداة التحكم وهذا العرض يقاس بالبكسل.

هـذه الخصـائص لا يمكـن الوصـول إلـيها فـى المكونات الغير مرئية ولكن على كل حـال تعمـل لغـة Delphi على تتبع الموضع الذى تضع فيه أيقونات المكونات بالفورم التى تعدهـا. ونود هنا القول بأنه فى أغلب الأحيان ستقوم بتحديد قيم هذه الخصائص عن طريق الـتعامل مـع صـورة أداة الـتحكم الموجـودة فى الفورمة أو عن طريق استخدام باللية الضبط Alignment Palette.

خصائص العرض Display Properties

فيما يلى مجموعة الخصائص التى تتحكم فى المظهر العام لأى أداة تحكم :

الخاصية	الوصف والاستخدام
Color	تغيير لون الخلفية لأداة التحكم.
Font	تغيير كل من لون ونوع ونمط وحجم الحروف.
Cursor	تحديـد الصـورة المسـتخدمة لتمثـيل مؤشـر المـاوس عـندما يمـر فوق المنطقة التى تحتلها أداة التحكم بالفورمة.
DesktopFont	تحديـد ما إذا كانت أداة التحكم تستخدم فونتات أيقونات الويندوز عند كتابة النصوص.



الخاصية DesktopFont تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.

خصائص الأب Parent Properties

لكـى تعمـل عـلى تحسـين وتطويـر المظهـر العـام للتطبيق الذى تتولى إعداده فإنك تسـتطيع إذن جعـل أى أداة تحكـم تشـبه الحاوية التى يوجد بداخلها–هذه الحاوية يطلق علـيها الأب Parent–وذلـك عـن طـريق تخصـيص القيمة المنطقية True للخصائص التالية والتى تخص الحاوية (الأب) :

الخاصية	الوصف والاستخدام
ParentColor	تحديد أين تبحث أداة التحكم عن معلومات اللون الخاصة بها.
ParentFont	تحديد أين تبحث أداة التحكم عن معلومات الفونت الخاصة بها.
ParentShowHint	تحديد أين تبحث أداة التحكم لكى تعثر على معلومات المساعدة الخاصة بها.

خاصية التجول Navigation Property

الخاصية التالية تعمل على تحديد الطريقة التى يتبعها المستخدمين للتطبيق للتجول بين أدوات التحكم الموجودة فى اى فورمة :

الخاصية	الوصف والاستخدام
Caption	هذه الخاصية تشتمل على سلسلة حرفية تعتبر عنوان label للمكون (أداة التحكم). هذا ولكى تجعل أى حرف فى هذه السلسلة الحرفية تحته خط (وفى هذه الحالة يعرف هذا الحرف بحرف الاستدعاء) ضع العلامة & قبل الحرف مباشرة. ومن ثم يستطيع المستخدم فى مرحلة التشغيل Run-Time التعامل مع المكون أو أداة التحكم عن طريق الضغط على هذا الحرف مع المفتاح Alt فى نفس الوقت.

خصائص السحب والإسقاط Drag-and-Drop

الخصائص التالية تؤثر بشكل مباشر على الطريقة التى تتم بها عملية السحب والإسقاط لأدوات التحكم :

الخاصية	الوصف والاستخدام
DragMode	هذه الخاصية تحدد الطريقة التى تبدأ بها عملية السحب. والقيمة الافتراضية لهذه الخاصية عبارة عن dmManual كما أن التطبيق ينبغى عليه استدعاء الأسلوب BeginDrag للبدء فى عملية

	السحب. أما عندما تكون قيمة هذه الخاصية عبارة عن dmAutomatic فى هذه الحالة تبدأ عملية السحب بمجرد أن الضغط على زر الماوس.
DragCursor	تعمل هذه الخاصية على تحديد شكل مؤشر الماوس عندما يمر فوق المكونات أو أدوات التحكم التى تقبل أن تجرى معها عملية السحب والاسقاط.



الخاصية DragCursor تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.

خصائص السحب والإرساء Drag-and-Dock

فيما يلى نقدم لك مجموعة الخصائص التى تتحكم فى سلوك عملية السحب والإرساء Drag-and-Dock التى تجرى على أدوات التحكم التى تنحدر من TWinControl :

الخاصية	الوصف والاستخدام
Floating	تحديد ما إذا كانت أداة التحكم عائمة أم لا.
DragKind	تحديد ما إذا كانت أداة التحكم قد تم سحبها بطريقة عادية أم تم سحبها لإرسائها بأحد جوانب الشاشة.
DragMode	تحديد طريقة البدء فى إجراء عملية السحب والإسقاط أو عملية السحب والإرساء على أداة التحكم.
FloatingDockSiteClass	تحديد قطاع أداة التحكم المؤقتة الذى يستضيف أداة التحكم عندما تكون عائمة وسط الشاشة.
DragCursor	عبارة عن شكل مؤشر الماوس الذى يظهر على الشاشة فى أثناء عملية السحب.

DockOrientation	تحديد طريقة ارساء أداة التحكم بالنسبة لأدوات التحكم الأخرى التى ترسو الآن فى الحاوية (الأب).
HostDockSite	تحديد أداة التحكم التى سيتم الإرساء بها.
LRDockWidth	تحديد عرض أداة التحكم عندما ترسو أفقيا.
TBDockHeight	تحديد ارتفاع أداة التحكم عندما ترسو رأسيا.
UnDockHeight	تحديد ارتفاع أداة التحكم عندما تكون عائمة وسط أداة التحكم الحاوية.
UnDockWidth	تحديد عرض أداة التحكم عندما تكون عائمة وسط أداة التحكم الحاوية.



هذه الخصائص السالفة الذكر تكون لأدوات التحكم الموجودة فى المكتبة VCL فقط.

الفصل الرابع

تمرين عملى
إنشاء تطبيق بسيط
لتحرير النصوص



مقدمة عامة

من خلال هذا التمرين العملى سنتعرف سويا على كيفية إنشاء محرر للنصوص متكامل يشتمل على مجموعة من القوائم وشريط أدوات وشريط حالة.



هذا التمرين العملى يمكن تنفيذه من خلال كافة إصدارات لغة Delphi كما أن التطبيق الناتج سيكون قادر على العمل من خلال بيئة الويندوز فقط.

البدء فى تطبيق جديد

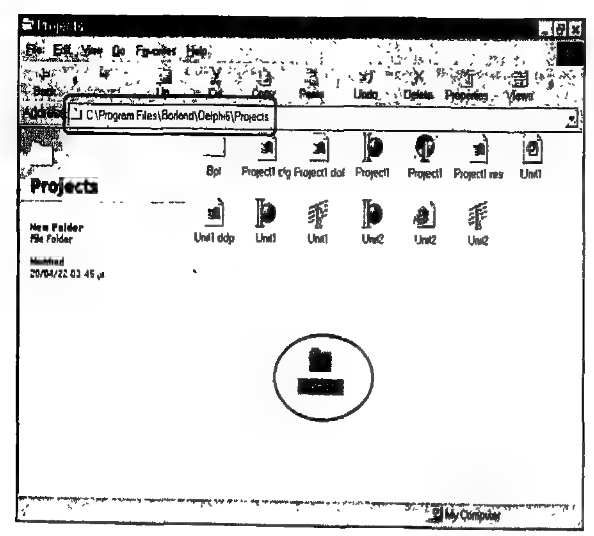
قبل فى تبدأ فى تطبيق جديد قم بإنشاء الفهرس الذى سيتم فيه تخزين ملفات التطبيق التى سيتم إنشاؤها من خلال Delphi وللقيام بذلك اتبع الخطوات التالية :

(١) قم بإنشاء فهرس يسمى TextEditor وذلك بالمسار التالى :

C:\Program Files\Borland\Delphi6\Projects

كما هو موضح بالشكل التالى :

شكل توضيحى :



(٢) افتح مشروع جديد بلغة Delphi وذلك من عن طريق فتح القائمة File ثم اختيار New ثم اختيار Application كما هو موضح بالشكل التالى :

شكل توضيحي :

Delphi 6 - Project1
File Edit Search View Project Run Component
New — Application
Open... — CLX Application
Open Project... Ctrl+F11 — Data Module
Reopen — Form
Save Ctrl+S — Frame
Save As... — Unit
Save Project As... — Other...
Save All Shift+Ctrl+S



كل تطبيق يتم التعامل معه على أساس أنه مشروع. وعندما تبدأ في التعامل مع لغة Delphi فإنها تقوم بشكل أفتراضى بإنشاء مشروع فارغ.



لو كان هناك مشروع آخر مفتوح بالفعل في هذه الحالة افتح القائمة File ثم اختر منها New ثم اختر Application كما ذكرنا سابقا.

عندما تفتح مشروع جديد تقوم لغة Delphi على الفور وبشكل تلقائي بإنشاء الملفات التالية :

- الملف Project1.dpr وهو عبارة عن ملف كود مصدرى يكون ملحقا بالمشروع. وهذا الملف يطلق عليه ملف المشروع.
- الملف Unit1.pas وهو عبارة عن ملف كود مصدرى يكون ملحقا بفورمة المشروع الأساسية. وهذا الملف يطلق عليه ملف الوحدة.
- الملف Unit1.dfm وهو عبارة عن ملف مصدرى يعمل على تخزين معلومات عن فورمة المشروع الأساسية. وهو يطلق عليه ملف الفورمة.

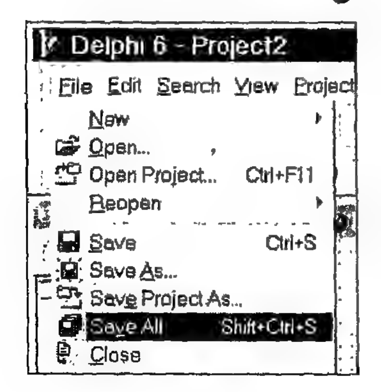


كل فورمة لديها كل من ملف الوحدة (Unit1.pas) وملف الفورمة (Unit1.dfm) الخاص بها. ولو أنك قمت بإنشاء فورمة ثانية سيكون لها كل من ملف الوحدة (Unit2.pas) وملف الفورمة (Unit2.dfm) الخاص بها وهكذا...ومثل هذه الملفات يتم إنشاؤها تلقائيا.



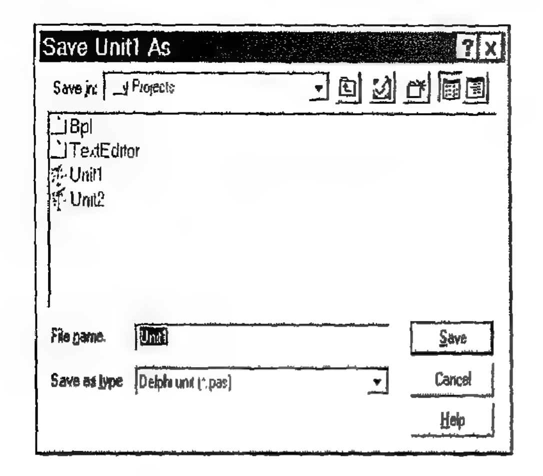
(٣) افتح القائمة File ثم اختر منها الأمر Save All كما هو موضح بالشكل التالى وذلك لكى تحفظ الملفات الخاصة بالمشروع الذى تعده :

شكل توضيحى :



(٤) والآن يظهر على الشاشة صندوق الحوار Save Unit1 As الموضح فى الشكل التالى :

شكل توضيحى :

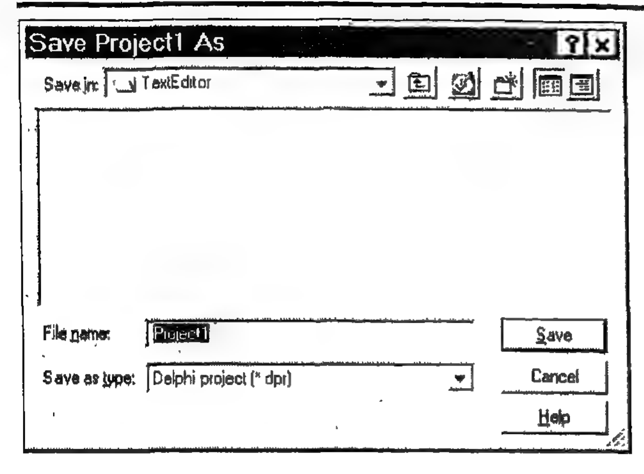


من خلال صندوق الحوار هذا قم بالآتى :

- الذهاب إلى المجلد TextEditor الذى قمت بإنشاؤه.
- احفظ الوحدة Unit1 مستخدما الاسم الافتراضى لها وهو Unit1.pas.

(٥) بعد ذلك يظهر على الشاشة صندوق الحوار Save Project1 As الموضح فى الشكل التالى :

شكل توضيحى :



من خلال صندوق الحوار هذا قم بالآتى :

- حفظ المشروع مستخدما الاسم (TextEditor.dpr).



الملف التنفيذى لهذا المشروع سيكون بنفس اسم المشروع ولكن بالامتداد .exe



فيما بعد ستتمكن من إعادة حفظ ما تقوم به من عمل عن طريق الأمر Save All بالقائمة File ولكن فى هذه المرة لن تظهر صناديق الحوار السالفة الذكر.

عندما تحفظ المشروع الذى تعده فإن Delphi تقوم بإنشاء عدد من الملفات الإضافية وتضعها فى الفهرس الذى أعددته للمشروع (الفهرس TextEditor). ومن بين هذه الملفات كل من الملف TextEditor.dof وهو عبارة عن ملف خيارات الديلفى والملف TextEditor.cfg وهو عبارة عن ملف التهيئة والملف TextEditor.res وهو ملف مصادر الويندوز.



لست فى حاجة لأن تقلق بخصوص هذه الملفات ولكن لا تحاول مسح هذه الملفات فهى ضرورية بالنسبة للمشروع الذى تتولى إعداده.



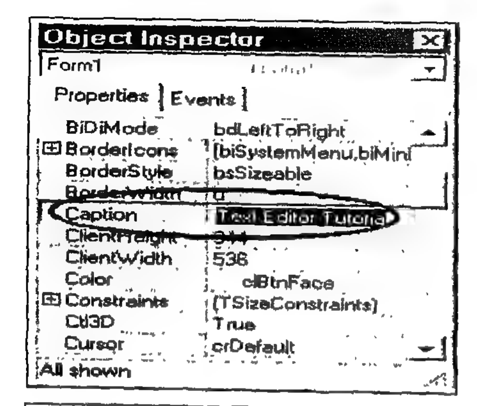
تحديد قيم الخصائص

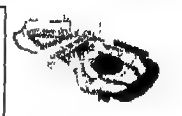
عندما تفتح مشروع جديد تقوم لغة Delphi بعرض الفورمة الأساسية للمشروع والتي تسمى Form1 (وهو اسم افتراضي). وأنت ستقوم بإنشاء واجهة الاستخدام User Interface والأجزاء الأخرى من التطبيق الذى تتولى إعداده وذلك عن طريق وضع المكونات بهذه الفورمة.

بجوار الفورمة ستشاهد نافذة أداة فحص الكائن Object Inspector والتي من خلالها تستطيع أن تحدد قيم خصائص الفورمة والمكونات التي تضعها بالفورمة. وعندما تحدد قيم الخصائص تقوم لغة Delphi بإعداد الكود البرمجي المصدري الخاص بهذا التحديد بدلا من أن تقوم بهذه المهمة بنفسك. ونود هنا القول بأن القيم التي تحددها في نافذة Object Inspector يطلق عليها قيم تحديدية لفترة التصميم Design-Time settings. وللقيام بذلك اتبع الخطوات التالية :

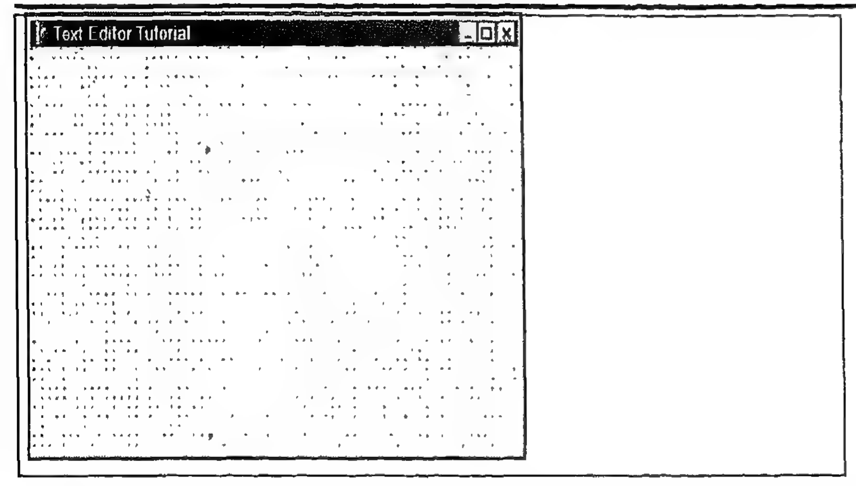
(١) في نافذة Object Inspector ابحث عن الخاصية Caption للفورمة وعندها اكتب Text Editor Tutorial ليكون بديل للقيمة الافتراضية Form1 لهذه الخاصية كما هو موضح بالشكل التالي :

شكل توضيحي :



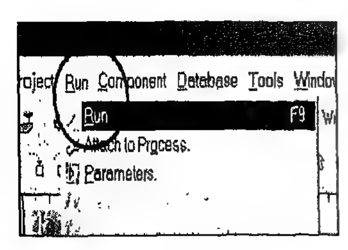


لاحظ أن القيمة التي يتم تخصيصها للخاصية Caption للفورمة تظهر في شريط العنوان Title bar لهذه الفورمة كما هو موضح بالشكل التالي :



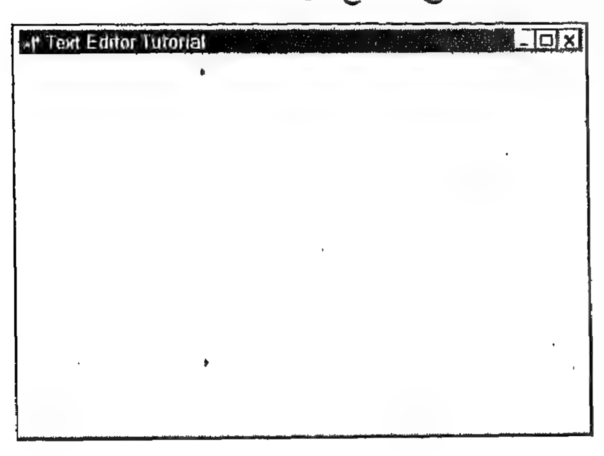
(٢) الآن قم بتشغيل الفورمة (حتى ولو كانت غير مشتملة على أى مكونات) عن طريق الضغط على المفتاح F9 بلوحة المفاتيح أو فتح القائمة Run واختيار الأمر Run منها كما هو موضح بالشكل التالى :

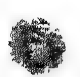
شكل توضيحى :



والآن تظهر الفورمة على الشاشة بالوضع الموضح فى الشكل التالى :

شكل توضيحى :

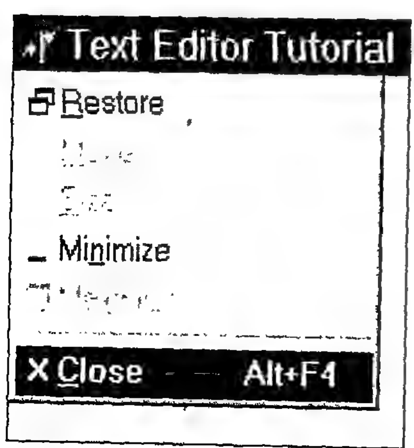




(٣) لكـى تعود مرة أخرى إلى مشهد الفورمة Form1 فى مرحلة التصميم قم بتنفيذ أى من الآتى :

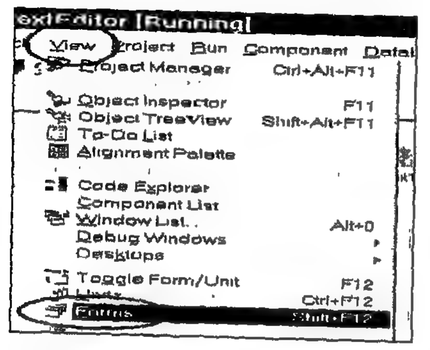
- انقـر بالماوس على الزر (×) فى الجانب الأيمن لشريط عنوان فورمة التطبيق (مشهد التشغيل runtime view للفورمة والموضح فى الشكل السابق).
- انقـر بالمـاوس عـلى الأيقونـة الموجـودة على يسار شريط العنوان لتظهر قائمة تحكم ومنـها اختر Close كما هو موضح بالشكل التالى :

شكل توضيحى :



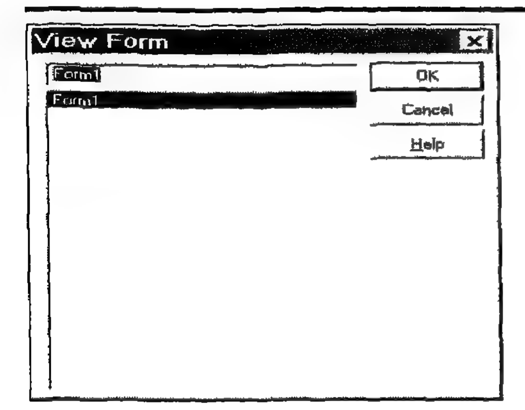
- افـتح القائمة View ثم اختر منها Forms ثم اختر Forms كما هو موضح بالشكل التالى (أو اضغط على المفتاحين Shift+F12 بلوحة المفاتيح):

شكل توضيحى :



ليظهر على الشاشة صندوق الحوار View Form الموضح فى الشكل التالى :

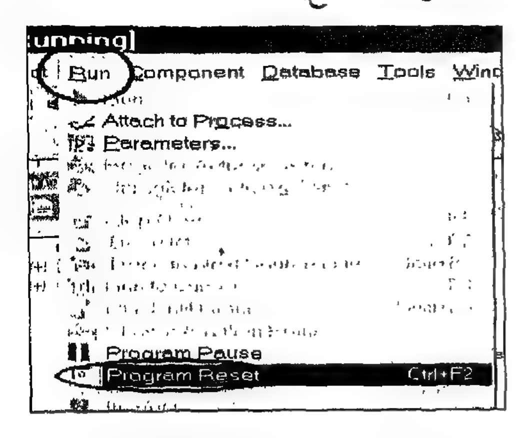
شكل توضيحى :



بصندوق الحوار هذا اختر Form1 ثم انقر بالماوس على المفتاح Ok.

- افـتح القائمـة Run واخـتر مـنها الأمـر Program Reset كمـا هـو موضح بالشكل التالـى (أو اضغط على المفتاحين Ctrl+F2 بلوحة المفاتيح):

شكل توضيحى :



إضافة المكونات إلى الفورمة

قـبل أن تـبدأ فـى إضـافة المكونـات إلى الفورمـة سـتحتاج للتفكير فى أبسط طريقة لإنشـاء واجهـة الاسـتخدام UI (اختصـار للمصـطلح User Interface) للتطبيق الذى تتولى إعـداده. وواجهـة الاسـتخدام UI هى التى تسمح للمستخدم الذى يتعامل مع التطبيق الذى تعـده بـأن يـتفاعل مـع العناصـر الـتى يـتألف مـنها التطبـيق وبالتالى يجب تصميم واجهـة الاستخدام UI بحيث تكون سهلة الاستخدام.



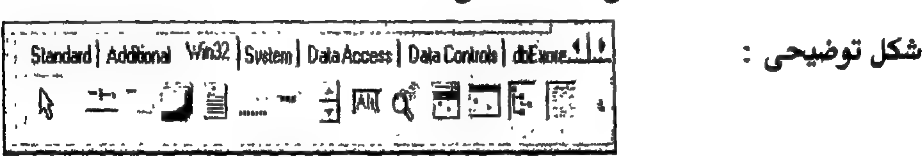
تتضمن لغة Delphi العديد من المكونات التى تمثل الأجزاء التى يتألف منها أى تطبيق يتم إعداده بلغة Delphi. فعلى سبيل المثال هناك مكونات (منحدره من كائنات) بباللية المكونات التى تجعل من السهولة بمكان إضافة العديد من العناصر المرئية والغير مرئية للبرنامج مثل القوائم Menus وشرائط الأدوات toolbars وصناديق الحوار وغيرها الكثير والكثير...

تطبيق محرر النصوص الذى نحن بصدده هنا يتطلب وجود منطقة لتحرير النصوص وشريط حالة Status bar لعرض بعض المعلومات مثل اسم الملف الذى يتم تحريره الآن. كما يحتاج أيضا لمجموعة من القوائم Menus. كذلك من الممكن أن يشتمل على شريط أدوات يضم مجموعة من الأيقونات لسهولة الوصول للأوامر الموجودة بالقوائم. هذا ونود هنا القول بأن جمال تصميم واجهة الاستخدام UI باستخدام لغة Delphi يتمثل فى أنك تستطيع التعامل مع العديد من المكونات المختلفة ومشاهدة النتائج المترتبة على ذلك على الفور. وبهذه الطريقة تستطيع بكل سرعة إعداد واجهة الاستخدام UI لأى تطبيق بأقل قدر من المجهود.

لكى تبدأ فى تصميم محرر النصوص عليك أن تضيف كل من المكون RichEdit والمكون StatusBar للفورمة وذلك من خلال الخطوات التالية :

(١) لكى تنشأ منطقة للنصوص عليك أولا أن تضيف المكون RichEdit. ولكى تعثر على هذا المكون انقر بالماوس على الصفحة Win32 بباللية المكونات لتظهر على السطح كما هو موضح بالشكل التالى :

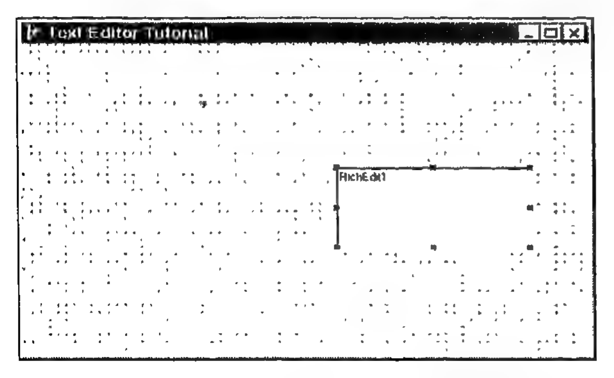
شكل توضيحى :



ثم ضع مؤشر الماوس على الأيقونة RichEdit . وعندما تعثر على المكون RichEdit فى هذه الحالة يمكن أن تقوم بأى من الآتى :

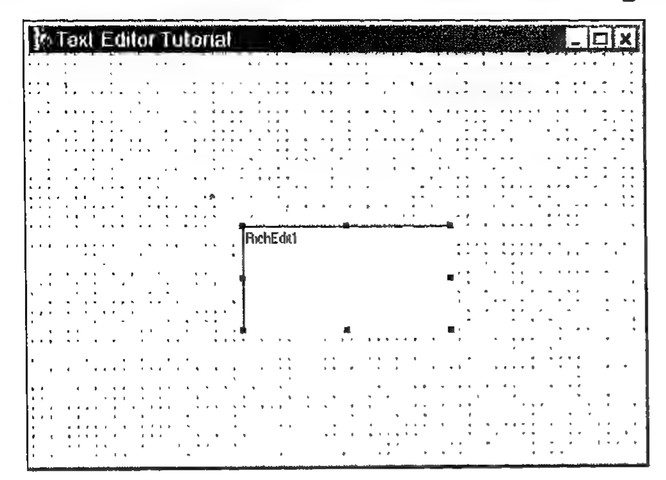
- تنقر بالماوس على أيقونة المكون RichEdit بالليتة المكونات ثم تنقر بالماوس على الموضع الذى ترغب فى وضع المكون به داخل الفورمة ليتم وضعه بهذا الموضع كما هو موضح بالشكل التالى :

شكل توضيحى :



- أو تنقر بالماوس نقرا مزدوجا على أيقونة المكون بالليتة المكونات ليتم وضعه فى مركز الفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :



كل مكون من المكونات المتاحة لدى لغة Delphi يعتبر قطاع كما أن وضع مكون فى فورمة يؤدى إلى إنشاء حالة من هذا القطاع. هذا وبمجرد أن يصبح المكون فى الفورمة تقوم لغة Delphi بتكوين الكود البرمجى الضرورى لتوجيه حالة الكائن عند تشغيل التطبيق. والشكل التالى يوضح لنا الكود البرمجى الذى تم تكوينه للمكون RichEdit :

شكل توضيحى :

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs;

type
  TForm1 = class(TForm)
    RichEdit1: TRichEdit;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation
```

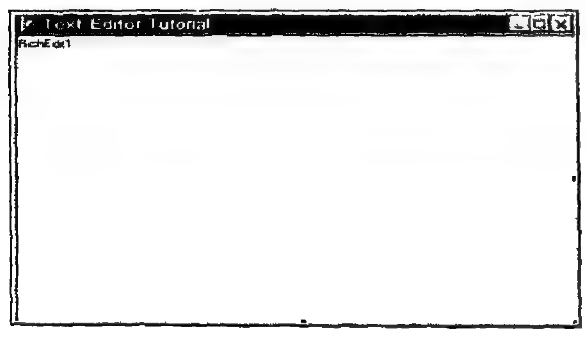
(٢) فـى أثـناء كون المكون RichEdit مختار (معلم عليه) بالفورمة قم بالبحث عن الخاصـية Align بالـنافذة Object Inspector ثـم افتح القائمة المنسدلة لهذه الخاصية واختر منها alClient كما هو موضح بالشكل التالى :

شكل توضيحى :

Object Inspector
RichEdit1
Properties | Events
Align: alClient
alBottom, alClient, alCustom, alLeft, alNone, alRight, alTop
Alignment
Anchors
BevelEdges
BevelInner
BevelKind
BevelOuter
BevelWidth
BiDiMode: bdLeftToRight
BorderStyle: bsSingle
BorderWidth: 0
All shown

تلاحظ الآن أن المكون RichEdit قد ملء الفورمة بأكملها كما هو موضح بالشكل التالى :

شكل توضيحى :



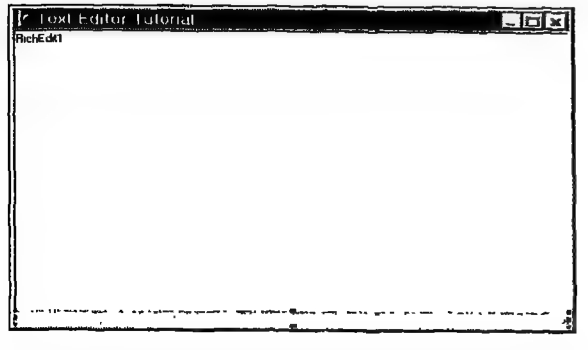
ومن ثم فقد أصبح لديك الآن منطقة كبيرة لتحرير النصوص.



باختيار القيمة alClient من القائمة المنسدلة للخاصية Align يصبح حجم أداة التحكم RichEdit قابل للتغيير ويتحدد على حسب حجم الفورمة الموجودة بها أداة التحكم RichEdit وذلك فى حالة كون الفورمة تتمتع بإمكانية تغيير الحجم.

(٣) انقر بالماوس نقرا مزدوجا على أيقونة المكون StatusBar بالصفحة Win32 بباليتة المكونات. مما يؤدى إلى إضافة شريط حالة الجزء السفلى للفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :

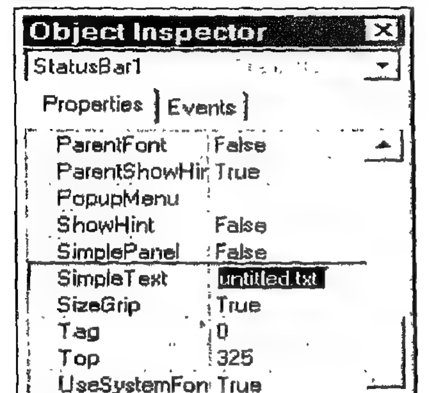


(٤) لكى تنشأ panel واحدة بشريط الأدوات لعرض اسم وموضع الملف الذى يتم تحريره من خلال محرر النصوص الذى نعده الآن عليك إذن القيام بالآتى :

- تأكد من أن شريط الحالة مختار (معلم عليه) فى الفورمة كما هو موضح بالشكل السابق.
- فى النافذة Object Inspector ابحث عن الخاصية SimpleText وعندها اكتب untitled.txt كما هو موضح بالشكل التالى :



شكل توضيحى :

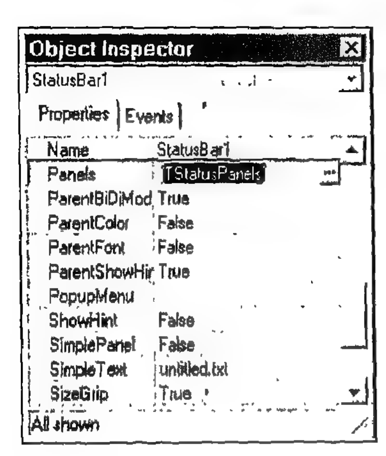




نتيجة ما سبق أنه عندما تستخدم محرر النصوص لو كان الملف الذى يتم تحريره لم يتم حفظه بعد فإن اسم الملف فى هذه الحالة سيكون untitled.txt.

- انقر بالماوس على الأيقونة (...) الموجودة بجوار القيمة TstatusPanel عند الخاصية Panels كما هو موضح بالشكل التالى :

شكل توضيحى :



ليظهر على الشاشة صندوق الحوار Editing StatusBar1.Panels الموضح فى الشكل التالى :

شكل توضيحى :

Editing StatusBar1.Panels

- بشـريط الأدوات الخـاص بصـندوق الحـوار هـذا انقـر بالماوس على الأيقونة New Action ليتم إضافة Panel بشريط الحالة كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing StatusBar1.Panels

0 - TStatusPanel

تستطيع أيضا الوصول لصندوق الحوار Editing StatusBar1.Panels عن طريق النقر بالماوس نقرا مزدوجا على شريط الحالة الموجود بالفورمة.

(٥) انقر بالماوس على الزر (×) لغلق صندوق الحوار Editing StatusBar1.Panels.

لقـد إنتهيـت الآن مـن إعـداد وتهيئة واجهة الاستخدام UI لمحرر النصوص الذى تتولى إعداده.

إضافة الدعم الخاص بالقوائم وشرائط الأدوات

لجعـل التطبيـق لديـة القـدرة عـلى عمل أى شىء فإنه يحتاج لقائمة ومجموعة من الأوامـر ومن الأفضل إضافة شريط أدوات لهذا التطبيق. وبالرغم من إنك تستطيع كتابة كود برمجى لكل أمر من الأوامر على حدة إلا إن لغة Delphi تعمل على توفير ما يعرف بمدير الفعـل Action Manager للمساعدة فى جعل الكود البرمجى كله مجتمعا فى موضع واحد وهـو مـا يعرف بمركزية الكود البرمجى centralize the code بالإضافة لإعداد قائمة صورة



image list لمركزية الصور لإضافتها للأوامر الموجودة بالقوائم وشريط الأدوات بالتطبيق الذى تتولى إعداده.

من الأنسب أن يتم تسمية الأفعال المتصلة بالأوامر الموجودة بالقائمة مستعينا بكل من اسم القائمة واسم الأمر. فعلى سبيل المثال الفعل المسمى FileExit يشير إلى الأمر Exit الموجود بالقائمة File وهكذا...

الجدول التالى يقدم أنواع الأفعال التى يحتاج إليها تطبيق محرر النصوص الذى تتولى إعداده :

القائمة	الأمر	أيقونة بشريط الأدوات؟	الوصف
File	New	Yes	إنشاء ملف جديد.
File	Open	Yes	فتح ملف سبق إنشاؤه لتحرير محتوياته.
File	Save	Yes	حفظ الملف الذى يتم التعامل معه حاليا.
File	Save As	No	حفظ ملف باستخدام اسم جديد (كذلك السماح لك ملف جديد باستخدام اسم يتم اقتراحه).
File	Exit	Yes	إنهاء التعامل مع محرر النصوص.
Edit	Cut	Yes	استقطاع النص المختار ووضعه فى لوحة الاقتباس clipboard الخاصة ببيئة الويندوز.
Edit	Copy	Yes	عمل نسخه من النص المختار ووضعها فى لوحة الاقتباس clipboard الخاصة ببيئة الويندوز.

Edit	Paste	Yes	وضـع الـنص المخـزن فـى لوحـة الاقتـباس clipboard الخاصـة ببيئة الويـندوز بالموضـع الموجـود بـه مؤشر الكتابة بمنطقة تحرير النص.
Help	Content s	No	عـرض شاشة محتويات نظام المساعدة والـتى مـن خلالهـا تسـتطيع الوصول لمواضيع المساعدة.
Help	Index	No	عرض شاشة دليل المساعدة.
Help	About	No	عـرض صـندوق حـوار يشـتمل عـلى معلومات حول التطبيق.

لمركزية كل من الكود البرمجى والصور فى مدير أى فعل فإنك تحتاج لإضافة محرر مدير الفعل Action Manager إلى المشروع الذى تعده وذلك من خلال الخطوات التالية :

(١) انقـر بالمـاوس على الصفحة Additional بباليتة المكونات لتظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :

(٢) بهذه الصفحة انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager لإسقاطه فى القورمة كما هو موضح بالشكل التالى :

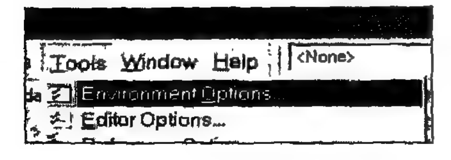
شكل توضيحى :



بسبب أن هذا المكون من المكونات التى تكون مختفية فى مرحلة التشغيل Run-Time لذلك تستطيع وضعه فى أى مكان بالفورمة.

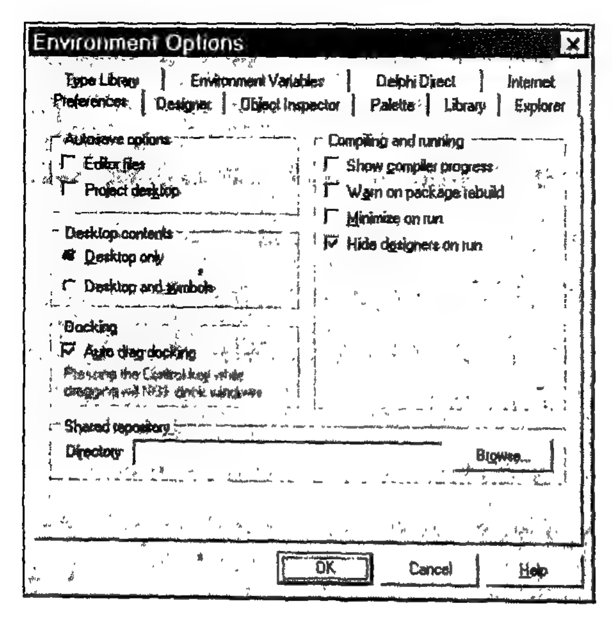
(٣) لعرض الخيارات الخاصة بالمكونات التى تكون مختفية فى مرحلة التشغيل Run-Time والتى تقوم بوضعها فى الفورمة افتح القائمة Tools واختر منها الأمر Environment كما هو موضح بالشكل التالى :

شكل توضيحى :



(٤) ليظهر على الشاشة صندوق الحوار Environment Options الموضح فى الشكل التالى :

شكل توضيحى :



(٥) انقر بالماوس على التبويب Designer ليظهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :

Environment Options
Type Library | Environment Variables | Delphi Direct | Internet
Preferences | Designer | Object Inspector | Palette | Library | Explorer
Grid options
Display grid
Snap to grid
Grid size
X 8 Y 8
Module creation options
New forms as text
Auto create forms & data modules
Options
Show component captions
Show designer hints
Show extended control hints
OK Cancel Help

(٦) بهـذا التبويـب عـلم بالماوس على الاختيار Show component captions ثم انقر بالماوس على المفتاح Ok.

إضافة الأفعال إلى مدير الأفعال Action Manager

فـى الـبداية ستقوم بإضافة الأفعال إلى مدير الأفعال كما ستحدد خصائص كل فعل مـن هـذه الأفعال. ومن المناسب فى هذه المرحلة أن تكون أسماء الأفعال التى تتصل بالأوامر الموجـودة بـالقوائم مؤلفة من اسم القائمة ومن اسم الأمر معا. فعلى سبيل المثال الفعل المسمى FileExit يشير إلى الأمر Exit الموجود بالقائمة File.

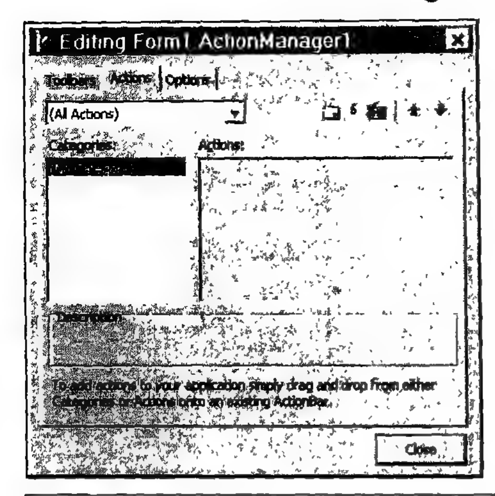
ستقوم الآن بإضافة نوعين من الأفعال الأولى سيتم تحديدخصائصها بنفسك والثانية تعـرف بأنهـا أفعـال قياسـية يـتم تحديـد خصائصـها تلقائيا. وللقيام بذلك اتبع الخطوات التالية :

(١) انقر بالمـاوس نقـرا مـزدوجا عـلى المكـون ActionManager الموجود بالفورمة ليظهر على الشاشة صندوق الحوار الموضح فى الشكل التالى :

شكل توضيحى :

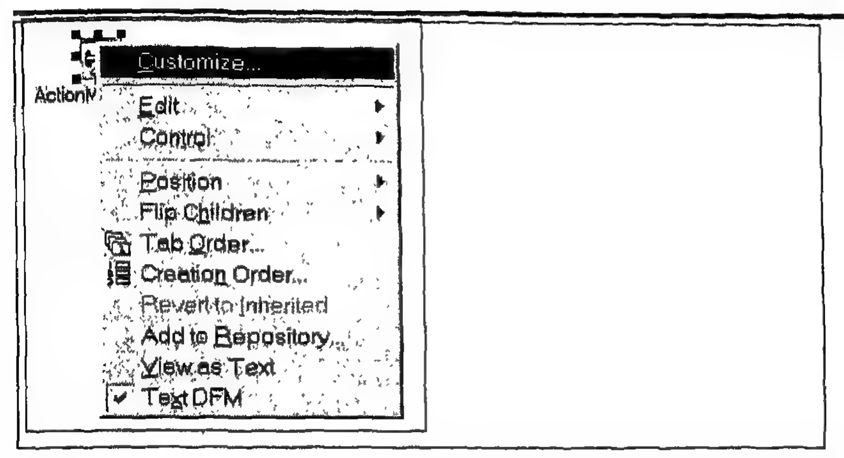
(٢) تـأكد مـن أن التبويـب Actions ظاهـر عـلى السطح وإلا فانقر عليه بالماوس لجعله ظاهر على السطح كما هو موضح بالشكل التالى :

شكل توضيحى :



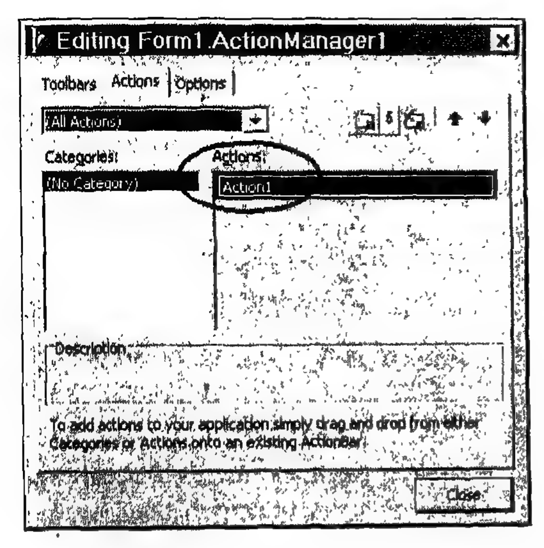


تستطيع أيضا النقر بالزر الأيمن للماوس على أيقونة المكون ActionManager بالفورمة ثم اختيار الأمر Customize من القائمة التى تظهر بجوار مؤشر الماوس كما هو موضح بالشكل التالى :



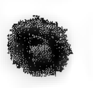
(٣) فـى أثـناء كـون No Category مخـتار فـى قائمـة العـرض Categories انقر بالماوس على الأيقونة New Action ليظهر Action1 فى قائمة العرض Actions كما هو موضح بالشكل التالى :

شكل توضيحى :

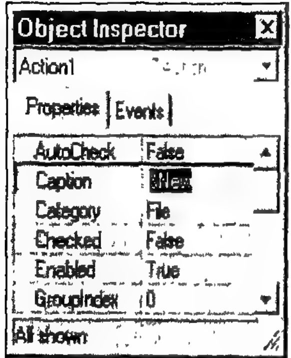


فى نافذة Object Inspector قم بتحديد الخصائص التالية للفعل Action1 :

- عند الخاصية Caption اكتب &New كما هو موضح بالشكل التالى :



شكل توضيحى :

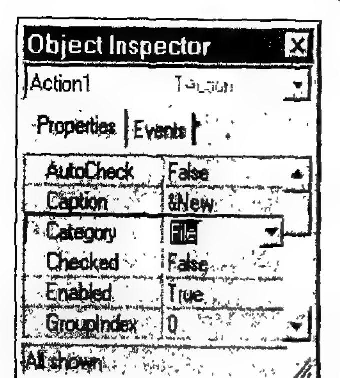




لاحظ أن كتابة العلامة & قبل حرف معين يؤدى إلى جعل هذا الحرف تحته خط فى مرحلة التشغيل وفى هذه الحالة يعرف هذا الحرف بأنه حرف الاستدعاء.

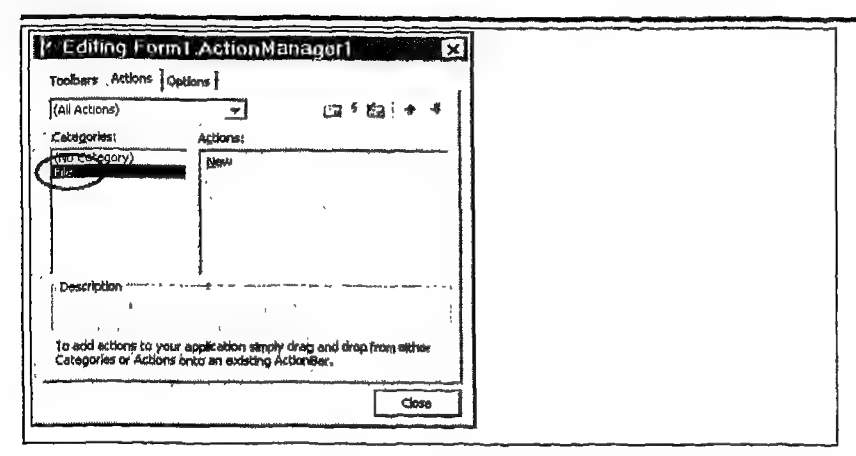
- عند الخاصية Category اكتب File كما هو موضح بالشكل التالى :

شكل توضيحى :



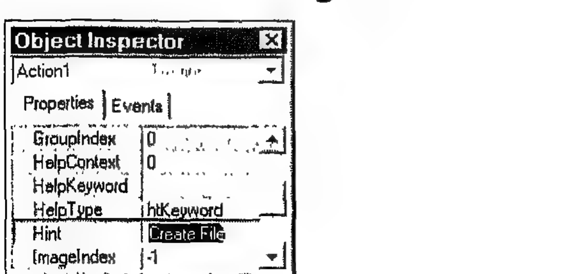


هذا التحديد يعمل على تنظيم الأوامر الخاصة بالقائمة File ووضعها فى موضع واحد. كما إنه يؤدى إلى ظهور File فى قائمة العرض Categories بصندوق الحوار كما هو موضح بالشكل التالى :



- عند الخاصية Hint اكتب Create file كما هو موضح بالشكل التالى :

شكل توضيحى :

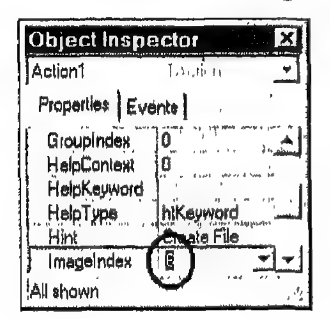




قيمة هذه الخاصية ستظهر فى أداة التعليق ToolTip التى تظهر عند وضع الماوس على الأمر New بالقائمة File لفترة بسيطة فى مرحلة التشغيل Run-Time.

- عند الخاصية ImageIndex اكتب القيمة ٦ كما هو موضح بالشكل التالى :

شكل توضيحى :



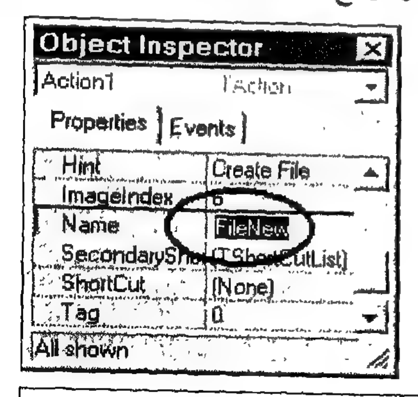




هذا التحديد سيربط الصورة رقم (٦) فى قائمة الصور ImageList (التى ستقوم بإعدادها) بهذا الفعل.

- عند الخاصية Name اكتب FileNew كما هو موضح بالشكل التالى :

شكل توضيحى :



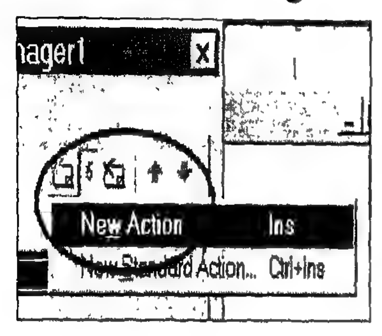


هذا الاسم سيكون اسم الفعل المرتبط بالأمر New الموجود بالقائمة File.

- الآن اضغط على المفتاح Enter بلوحة المفاتيح لحفظ القيم التى تم تخصيصها لهذه الخصائص.

(٤) بصندوق الحوار Action Manager انقر بالماوس على السهم المجاور للأيقونة New Action ثم اختر New Action كما هو موضح بالشكل التالى :

شكل توضيحى :



(٥) انقر بالماوس على Action1 فى قائمة العرض Actions ثم فى نافذة Object Inspector قم بتحديد قيم الخصائص التالية :



- عند الخاصية Caption اكتب &Save.
- عند الخاصية Category افتح القائمة المنسدلة واختر منها File..
- عند الخاصية Hint اكتب Save File.
- عند الخاصية ImageIndex اكتب القيمة ٨.
- عند الخاصية Name اكتب FileSave (للأمر Save بالقائمة File).

(٦) كرر الخطوة رقم (٤).

(٧) كرر الخطوة رقم (٥) ولكن مع الخصائص التالية :

- عند الخاصية Caption اكتب &Index.
- عند الخاصية Category اكتب Help.
- عند الخاصية Name اكتب HelpIndex (ليكون اسم الفعل الذى يشير للأمر Index الموجود بالقائمة Help).

(٨) كرر الخطوة رقم (٤).

(٩) كرر الخطوة رقم (٥) ولكن مع الخصائص التالية :

- عند الخاصية Caption اكتب &About.
- عند الخاصية Category افتح القائمة المنسدلة واختر منها Help.
- عند الخاصية Name اكتب HelpAbout (ليكون اسم الفعل الذى يشير للأمر About الموجود بالقائمة Help).

(١٠) لا تقم بإغلاق صندوق الحوار Action Manager.

إضافة الأفعال القياسية لقائمة الأفعال

فيما يلى ستقوم بإضافة الأفعال القياسية (الفتح Open والحفظ بأسم Save As وإنهاء البرنامج Exit والقص Cut والنسخ Copy واللصق Paste ومحتويات نظام المساعدة Help Contents) إلى مدير الأفعال Action Manager. وللقيام بذلك اتبع الخطوات التالية :

(١) ينبغى أن يكون مدير الأفعال Action Manager لا يزال مفتوحا ومعروضا على الشاشة. وإذا لم يكن كذلك انقر بالماوس نقرا مزدوجا على المكون ActionManager بالفورمة.

(٢) انقر بالماوس على السهم المجاور للأيقونة New Action ثم اختر New Standard Action كما هو موضح بالشكل التالى :

شكل توضيحى :





يمكن أيضا أن تستخدم الاختيار **New Standard Action** بالضغط على المفتاحين **Ctrl+Ins** بلوحة المفاتيح.

ليظهر على الشاشة صندوق الحوار Standard Actions Classes الموضح فى الشكل التالى :

شكل توضيحى :

Standard Action Class...
Available Action Classes:
(No Category)
TAction
Edit
TEditCut
TEditCopy
TEditPaste
TEditSelectAll
TEditUndo
TEditDelete
Format
TRichEditBold
TRichEditItalic
TRichEditUnderline
TRichEditStrikeOut
TRichEditBullets
TRichEditAlignLeft
TRichEditAlignRight
TRichEditAlignCenter
OK Cancel Help

(٣) فـى صندوق الحوار Standard Actions Classes تحرك داخل قائمة العرض حـتى تصـل إلى القسـم Edit ثـم اخـتر كـل مـن TEditCut وTEditCopy و TEditPaste (فـى اثـناء ذلـك اسـتخدم المفتاح Ctrl بلوحة المفاتيح عند النقر عـلى هذه الأفعال بالماوس ليتم إختيارهم معا) ثم انقر بالماوس على المفتاح Ok ليـتم إضـافة هـذه الأفعال إلى القسم Edit الجديد بقائمة العرض Categories بصندوق الحوار Action Manager كما هو موضح بالشكل التالى :

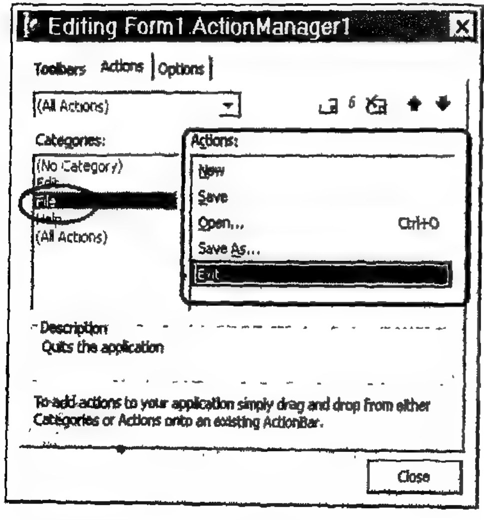
شكل توضيحى :

(٤) كـرر الخطـوة رقـم (٢) مـرة أخرى لفتح صندوق الحوار Standard Actions Classes.

(٥) بصـندوق الحـوار هـذا ابحـث عـن القسم File وعندما تصل إليه اختر الأفعال TFileOpen وTFileSaveAs وTFileExit ثـم انقـر بالماوس على المفتاح Ok لـتجد أنـه تم إضـافة هذه الأفعال إلى القسم File فى صندوق الحوار Action Manager كما هو موضح بالشكل التالى :



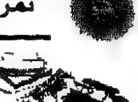
شكل توضيحى :

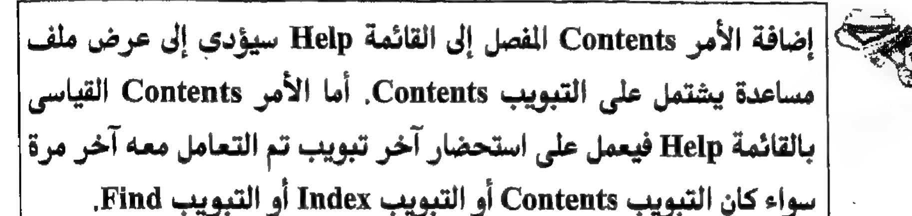


(٦) مـرة أخـرى كـرر الخطوة رقم (٢) مرة أخرى لفتح صندوق الحوار Standard Actions Classes.

(٧) بصـندوق الحـوار هـذا ابحـث عـن القسم Help وعندما تصل إليه اختر الفعل THelpContentsثـم انقـر بالمـاوس عـلى المفتاح Ok لتجد أنه تم إضافة هذه الأفعـال إلى القسم Help فى صندوق الحوار Action Manager كما هو موضح بالشكل التالى :

شكل توضيحى :





إضافة الأمر Contents المفصل إلى القائمة Help سيؤدى إلى عرض ملف مساعدة يشتمل على التبويب Contents. أما الأمر Contents القياسى بالقائمة Help فيعمل على استحضار آخر تبويب تم التعامل معه آخر مرة سواء كان التبويب Contents أو التبويب Index أو التبويب Find.

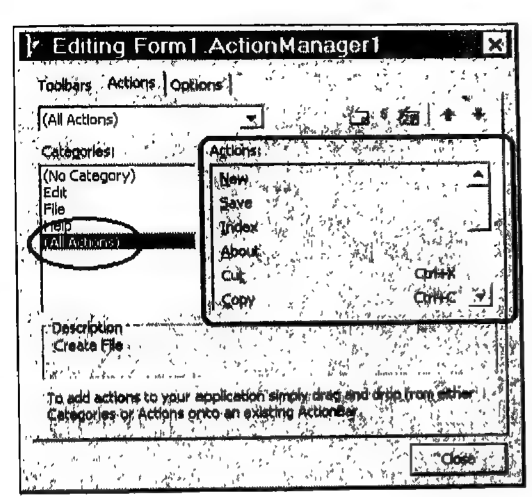
حتى الآن إنتهيت من إضافة كافة الأفعال القياسية التى يحتاج إليها تطبيق محرر النصوص الذى نتولى إعداده.



الأفعال القياسية لديها الخصائص الخاصة بها والتى يتم تحديد قيمتها تلقائيا بما فيها الخاصية ImageIndex.

(٨) انقر بالماوس على (All Actions) بالقسم Categories لتظهر فى القائمة Actions كافة الأفعال التى اضفتها سواء كانت قياسية أو غير قياسية كما هو موضح بالشكل التالى :

شكل توضيحى :



(٩) انقر بالماوس على المفتاح Close لغلق صندوق الحوار Action Manager.

(١٠) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها بالمشروع.



إضافة الصور لقائمة الصور Image List

فـى هذا الجزء من الفصل ستقوم بإضافة صور إلى مدير الأفعال من أجل استخدامها بكل من القوائم وشريط الأدوات.

الأفعـال القياسـية يكون ملحق بها صورا سابقة التخصيص وهذه الصور مستمدة من قائمـة صـور أساسـية Built-In تـأتى مـع لغـة Delphi. فعـلى سـبيل المـثال دليل الصورة Image Index للفعل القياسى Cut بالقائمة Edit عبارة عن 0.

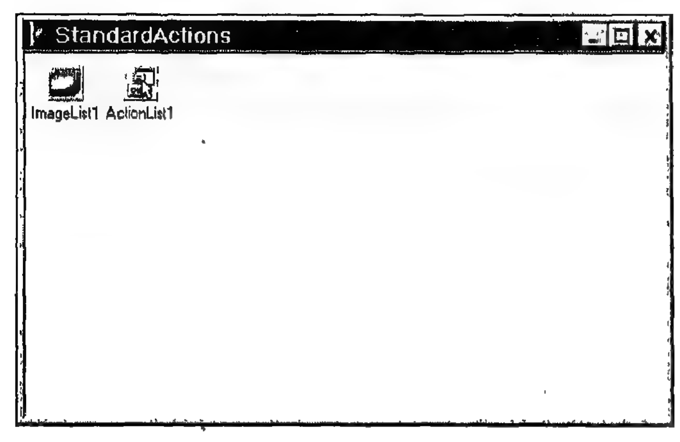


كافة الصور التى ستستخدمها لأوامر محرر النصوص الذى تتولى إعداده موجودة بقائمة الصور الأساسية السالفة الذكر.

لكى تضيف قائمة الصور اتبع الخطوات التالية :

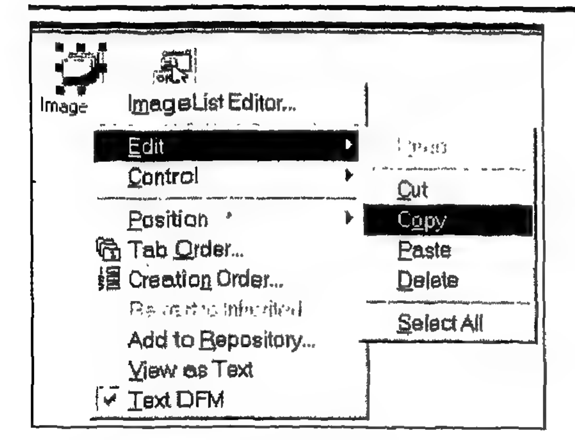
(١) لـو أنك قمت بتركيب لغة Delphi فى الموضع الذى يفترضه لك صندوق حوار التركيـب والتهيـئة فـى أثـناء عملـية التهيـئة فـى هـذه الحالة قم بفتح الملف ActnRes.pas الموجـــود بالمســـار C:\Program files\Borland\Delphi6\Source\VCL ليـتم فـتح نـافذة الأفعـال القياسـية StandardActions الموضح فى الشكل التالى :

شكل توضيحى :



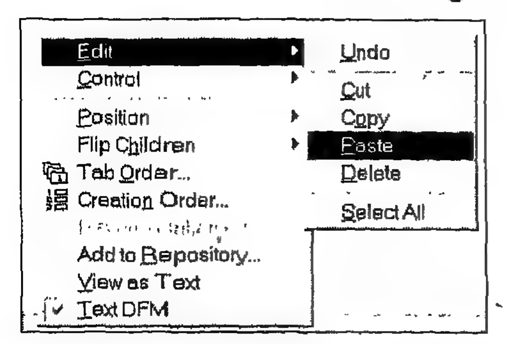
(٢) اخـتر المكـون ImageList1 ثم خذ منه نسخة بالنقر بالزر الأيمن على أيقونة هذا المكون ثم اختر Edit ثم اختر Copy كما هو موضح بالشكل التالى :

شكل توضيحى :



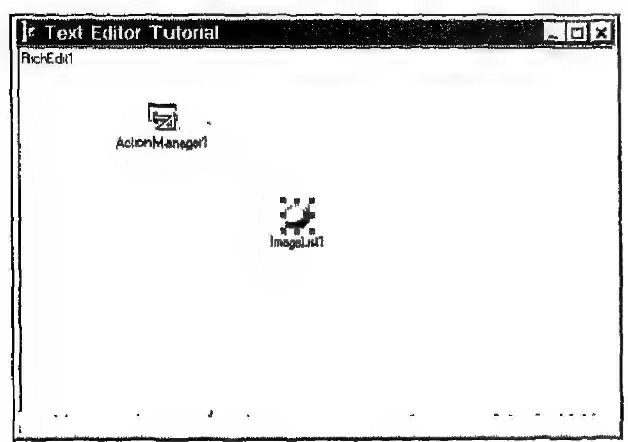
وفى الفورمة الخاصة بالمشروع الذى تعده انقر بالزر الأيمن للماوس بأى موضع بالفورمة ثم اختر Edit ثم اختر paste كما هو موضح بالشكل التالى :

شكل توضيحى :



ليتم وضع المكون ImageList1 بالفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :

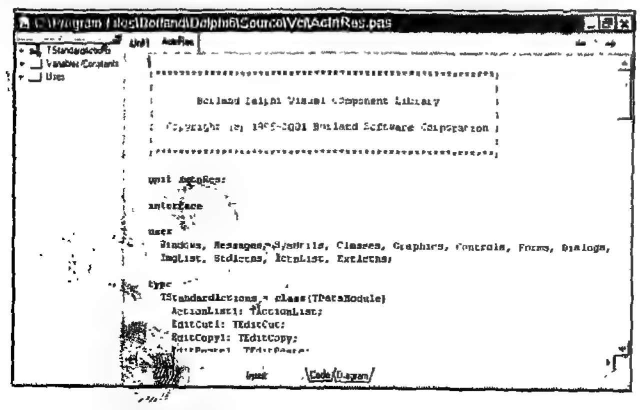




المكون ImageList1 يعتبر من طائفة المكونات التى تكون مختفية فى مرحلة التشغيل Run-Time ومن ثم فلا يهم الموضع الذى تختاره لكى تلصق فيه نسخة هذا المكون.

ستجد الآن أنه تم إضافة الوحدة ActnRes.pas إلى محرر الكود البرمجى كما هو موضح بالشكل التالى :

شكل توضيحى :



(٣) قم الآن بغلق النافذة StandardActions.

(٤) انقر بالماوس نقرا مزدوجا على الأيقونة ImageList1 الموجودة بالفورمة ليظهر على الشاشة صندوق الحوار ImageList (الموضح فى الشكل التالى) والذى يضم كافة الصور المتاحة والتى يمكن أن تستخدمها :

شكل توضيحى :

فيما يلى الجدول التالى يقدم لنا أرقام الصور المستخدمة مع كل أمر من الأوامر التى أضفتها لمحرر النصوص الذى تتولى إعداده :

القائمة	الأمر	قيمة الخاصية ImageIndex
Edit	Cut	0
Edit	Copy	1
Edit	Paste	2
File	New	6
File	Open	7
File	Save	8
File	Save As	30
File	Exit	43
Help	Contents	40



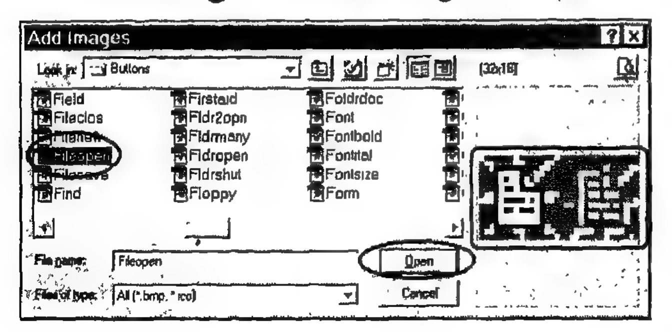
تستطيع أن تضيف صور من أى مجموعة مختلفة تماما. ففى صندوق الحوار Form1.ImageList انقر بالماوس على المفتاح Add ليظهر على الشاشة صندوق الحوار Add Images الموضح فى الشكل التالى :

من خلال صندوق االحوار هذا ابحث عن الفهرس المسمى Buttons الموجود بنفس المسار الذى تم تركيب لغة Delphi به علما بأن الموضع الافتراضى لهذا الفهرس عبارة عن الآتى :

C:\Program Files\Common Files\Borland Shared\Images\Buttons.

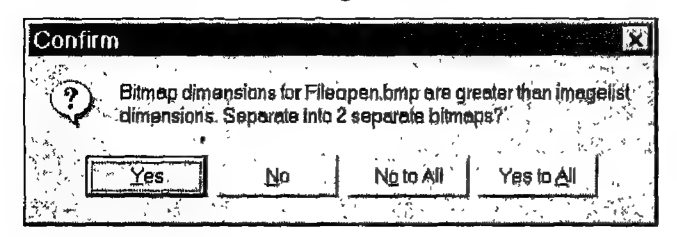
فعلى سبيل المثال لو أردت أن تخصص صورة من الصور الموجودة بالمجلد Buttons للأمر Open الموجود بالقائمة File (بالتطبيق الذى تعده) في هذه الحالة انقر بالماوس على الملف المسمى fileopen ثم على المفتاح Open كما هو موضح بالشكل التالى :

شكل توضيحى :



ليظهر على الشاشة صندوق الحوار Confirm الموضح فى الشكل التالى :

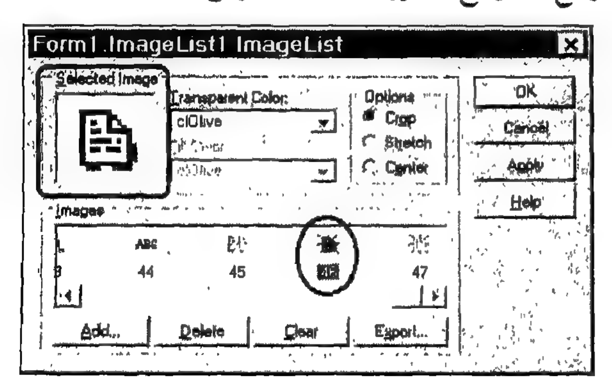
شكل توضيحى :



ليسألك عما إذا كنت ترغب فى تقسيم هذه الصورة إلى جزئين وفي حالتنا هذه انقر بالماوس على المفتاح Yes.

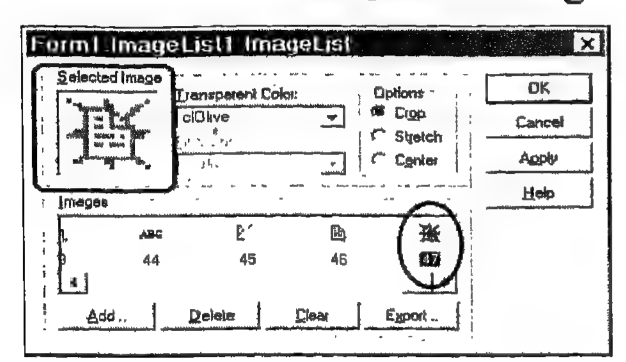
كل صورة من الصور لها وضع وهى نشطة Active ووضع آخر وهى غير نشطة. فعلى سبيل المثال الشكل التالى يوضح لنا وضع الصورة Fileopen وهى نشطة :

شكل توضيحى :



أما الشكل التالى فيوضح لنا وضع نفس الصورة وهى غير نشطة Grayed out :

شكل توضيحى :



والآن امسح وضع الصورة فى حالة كونها غير نشطة Grayed Out (آخر صورة) والذى بالنقر عليها بالماوس ثم على المفتاح Delete كما هو موضح بالشكل التالى :

شكل توضيحى :

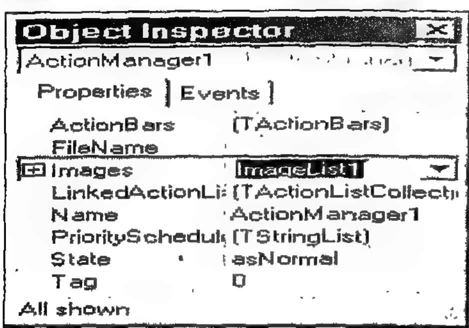
Form1.ImageList1 ImageList
Selected Image
Transparent Color:
clOlive
Options
Crop
Stretch
Center
OK
Cancel
Apply
Help
Images
43
44
45
46
Add...
Delete
Clear
Export...

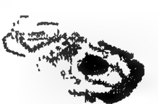
وبعد ذلك تأكد من أن دليل الصورة فى النافذة Object Inspector يتطابق مع الرقم الجديد المخصص لهذه الصورة فى القائمة التى تضم مجموعة الصور.

(٥) انقر بالماوس على المفتاح Ok لغلق صندوق الحوار ImageList.

(٦) اختر المكون ActionManager الموجود بالفورمة ثم عند الخاصية Images الخاصة به افتح القائمة المنسدلة واختر منها ImageList1 كما هو موضح بالشكل التالى :

شكل توضيحى :





بسبب أنك قمت بالفعل بتحديد قيمة الخاصية ImageIndex لكافة الأفعال التى أضفتها لهذا التطبيق لذلك فإن كل صورة يتم إضافتها تلقائيا إلى الفعل الصحيح المناظر لها. ومن ثم فقد تم إلحاق ٨ صور للـ٨ أفعال الموجودة بالتطبيق.

(٧) لكى تشاهد الصور الملحقة أو المرتبطة فى مدير الأفعال قم بفتح المكون Action Manager الموجود بالفورمة وتأكد من أن التبويب Actions هو الذى يظهر على السطح ثم انقر بالماوس على القسم All Actions بقائمة العرض Categories لتشاهد صورة صغيرة بجوار كل فعل كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing Form1 ActionManager1

Toolbars | Actions | Options

(All Actions)

Categories: (No Category), Edit, File, Help, (All Actions)

Actions: Cut Ctrl+X, Copy Ctrl+C, Paste Ctrl+V, Open... Ctrl+O, Save As..., Exit

Description
Create File

To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.

Close

(٨) والآن افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها بالمشروع.



يمكن أيضا أن تضغط على مجموعة المفاتيح Shift+Ctrl+S بلوحة المفاتيح معا فى نفس الوقت للقيام بنفس المهمة.

(٩) اجعل صندوق الحوار Action Manager مفتوحا.

الآن أنت على أهبة الاستعداد لكى تضيف القوائم وشريط الأدوات للمشروع.

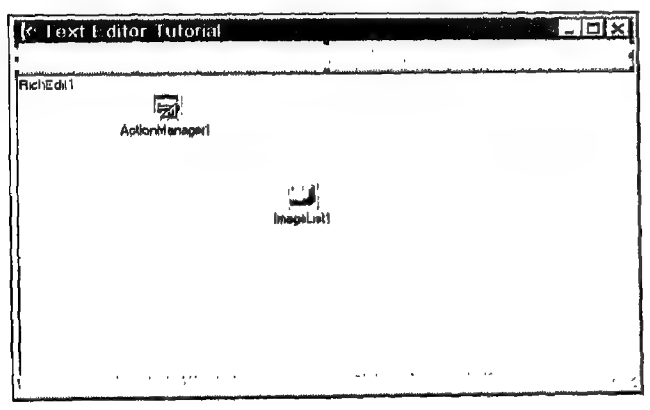
إضافة قائمة Menu للمشروع

فى خلال هذا الجزء والجزء القادم بهذا الفصل ستقوم بإضافة قائمة قابلة للتفصيل وشريط أدوات إلى المشروع الذى تعده وكلاهما يطلق عليه Action Bands.

شريط القائمة الأساسية يكون متضمنا ثلاثة قوائم وهى القائمة File والقائمة Edit والقائمة Help وكل منها تشتمل على الأوامر الخاصة بها. هذا ومع صندوق الحوار Action Manager تستطيع أن تسحب قسم كل قائمة (من قائمة العرض Categories) وما تحتويه من أوامر وإسقاطة فى شريط القائمة (كل ذلك فى خطوة واحدة فقط). وللقيام بذلك اتبع الخطوات التالية :

(١) من الصفحة Additional بباليتة المكونات انقر بالماوس نقرا مزدوجا على المكون ActionManinMenuBar  ليتم أضافتها إلى الفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :



(٢) افتح صندوق الحوار Action Manager إذا لم يكن مفتوح بالفعل ثم انقر بالماوس على القسم File في قائمة العرض Categories. وستلاحظ أن مجموعة الأوامر الخاصة بالقسم غير مرتبة بالترتيب الذى تطلبه بالضبط. ولكن على كل حال تستطيع بكل سهولة أن تقوم بتغيير هذا الترتيب وذلك عن طريق استخدام كل من الأيقونة Move Up (أو الضغط على المفتاحين Ctrl+Pgup بلوحة المفاتيح) والأيقونة Move Down (أو الضغط على المفتاحين Ctrl+PgDown بلوحة المفاتيح).

- اختر الفعل Open ثم انقر بالماوس على الأيقونة Move Up كما هو موضح بالشكل التالى :

شكل توضيحي :

Editing Form1.ActionManager1

Toolbars Actions Options

(All Actions)

Categories: (No Category) Edit File Help (All Actions)

Actions: New Save Open... Ctrl+O Save As... Exit

Description
Opens an existing file

To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.

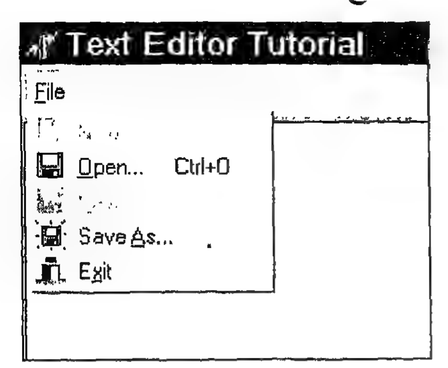
Close

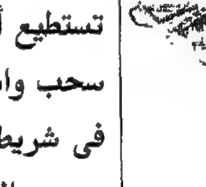
ومن ثم اجعل الأفعال الموجودة فى القسم File مرتبة كالآتى : New ثم Open ثم Save ثم Save As ثم Exit كما هو موضح بالشكل التالى :

شكل توضيحى :

(٣) اسحب القسم File من قائمة العرض Categories بصندوق الحوار Action Manager ثم اسقطه فى شريط القائمة لتجد أنه قد ظهرت القائمة File بالأوامر الخاص به بشريط القائمة كما هو موضح بالشكل التالى :

شكل توضيحى :



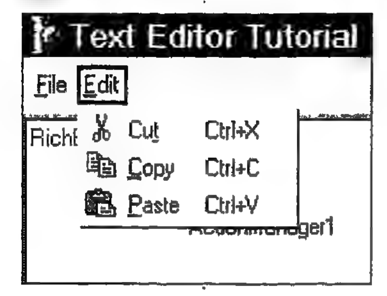


تستطيع أيضا أن تعيد ترتيب الأوامر بالقوائم وذلك بعد أن تنتهى من سحب واسقاط القوائم بشريط القائمة. فعلى سبيل المثال انقر على File فى شريط القائمة ومن ثم تظهر الأوامر الموجودة بهذه القائمة ثم قم بسحب الأمر Open ليصبح فوق الأمر New وهكذا...

(٤) من قائمة العرض Categories بصندوق الحوار Action Manager اسحب Edit واسقطه فى شريط القائمة بالفورمة كما هو موضح بالشكل التالى :

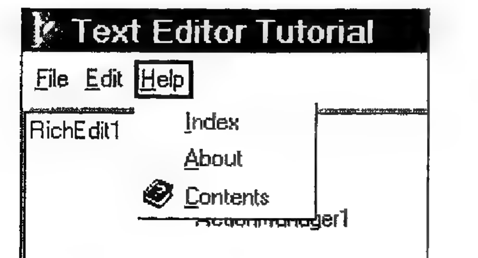


شكل توضيحى :



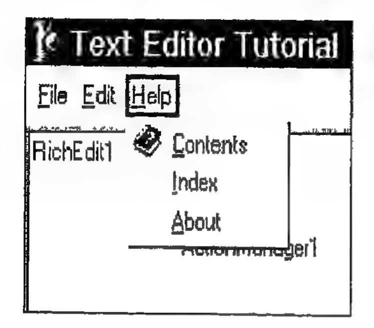
(٥) من قائمة العرض Categories بصندوق الحوار Action Manager اسحب Help واسقطه فى شريط القائمة بالفورمة كما هو موضح بالشكل التالى :

شكل توضيحى :



(٦) انقر بالماوس على القائمة Help بشريط القائمة لكى تشاهد محتوياتها ثم اسحب الأمر Contents ليصبح فوق الأمر Index كما هو موضح بالشكل التالى :

شكل توضيحى :



(٧) اضغط على المفتاح Esc بلوحة المفاتيح أو انقر بالماوس على Help مرة أخرى لغلقها.

(٨) افـتـح القائمـة File ثـم اخـتـر منها الأمر Save All ليتم حفظ التغييرات التي أجريتها بالمشروع.

والآن سـتحتاج لأن تضـيف شـريط أدوات لـتوفير طـريقة سـهلة للوصـول للأوامـر الموجودة بمجموعة القوائم الموجودة بالمشروع.

إضافة شريط أدوات toolbar للمشروع

بما إنك قمت بإعداد الأفعال في صندوق الحوار Action Manager لذلك تستطيع إضـافة بعـض مـن نفـس هـذه الأفعـال والتي كانت مستخدمه بمجموعة القوائم إلى عدد من الأيقونات بشريط أدوات toolbar.

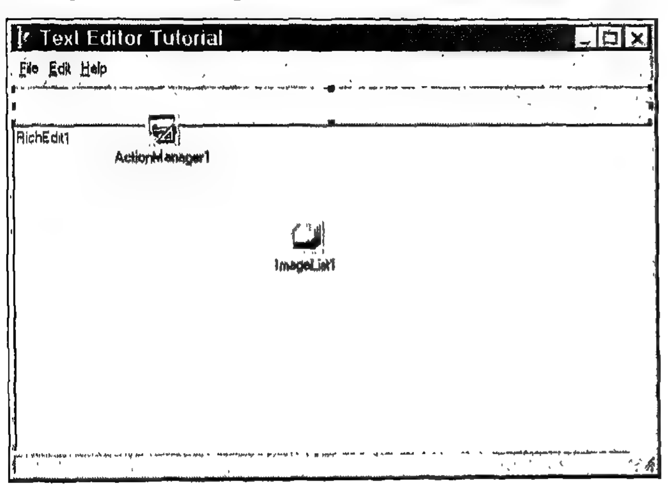


بعد أن تنتهي من شريط الأدوات هذا ستجد أنه مشابهه إلى حد كبير لشريط أدوات برامج الأوفيس ٢٠٠٠.

وللقيام بذلك اتبع الخطوات التالية :

(١) فـي الصفحة Additional بباليتة المكونات انقر بالماوس نقرا مزدوجا على أيقونة المكـون ActionToolBar ليـتم إضـافته إلى الفورمـة وهـو عبارة عن شريط أدوات فارغ يظهر اسفل شريط القائمة مباشرة كما هو موضح بالشكل التالي :

شكل توضيحي :







تستطيع أيضا إضافة شريط أدوات نت النوع action band وذلك عن طريق فتح صندوق الحوار Action Manager وجعل التبويب Toolbars يظهر على السطح كما هو موضح بالشكل التالى :

Editing Form1.ActionManager1

Toolbars | Actions | Options

Toolbars: (Checkmark toggles visibility)

New...

Delete

Toolbar Options

Caption Options

Selective

Apply caption options to all toolbars

Close

ثم انقر بالماوس على المفتاح New ليتم إضافة شريط أدوات جديد بالفورمة كما يظهر اسم شريط الأدوات الجديد فى قائمة العرض Toolbars بصندوق الحوار Action Manager كما هو موضح بالشكل التالى :

Editing Form1.ActionManager1

Toolbars | Actions | Options

Toolbars: (Checkmark toggles visibility)

ActionToolBar2

New...

Delete

Toolbar Options

Caption Options

Selective

Apply caption options to all toolbars

Close

(٢) إذا لم يكن صندوق الحوار Action Manager غير معروض على الشاشة قم بفتحه ثم اختر File بالقائمة Categories كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing Form1.ActionManager1

Toolbars Actions Options

(All Actions)

Categories: (No Category) Edit File Help (All Actions)

Actions: New Open... Ctrl+O Save Save As... Exit

Description
Create File

To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.

Close

(٣) فى القائمة Actions اختر كل من New وOpen وSave وExit كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing Form1.ActionManager1

Toolbars Actions Options

(All Actions)

Categories: (No Category) Edit File Help (All Actions)

Actions: New Open... Ctrl+O Save Save As... Exit

Description
Quits the application

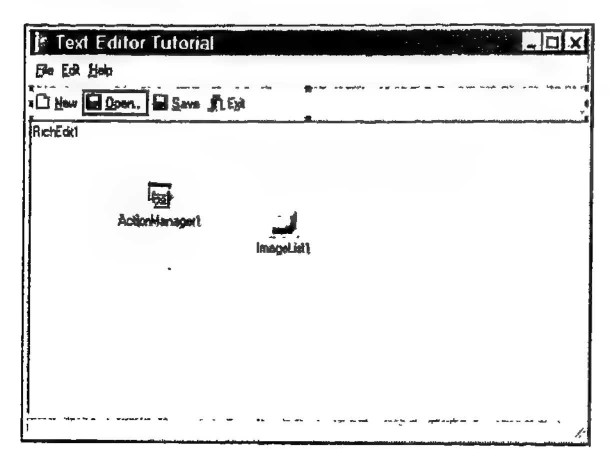
To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.

Close



ثم اسحب هذه العناصر بالماوس واسقطهم فى شريط الأدوات الموجود بالفورمة لتجد أن هـذه العناصر قـد ظهـرت تلقائـيا على شكل أيقونات فى شريط الأدوات كما هو موضح بالشكل التالى :

شكل توضيحى :



(٤) فـى صـندوق الحوار Action Manager اختر Edit بقائمة العرض Categories لـتظهر الأفعـال الخاصـة بالقسم Edit بقائمة العرض Actions كما هو موضح بالشكل التالى :

شكل توضيحى :

- فـى القائمـة Actions اخـتر كـل من Cut وCopy وPaste كما هو موضح بالشكل التالى :

شكل توضيحى :

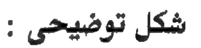
Editing Form1.ActionManager1
Toolbars Actions Options
(All Actions)
Categories:
(No Category)
Edit
File
Help
(All Actions)
Actions:
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Description
Inserts Clipboard contents
To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.
Close

- اسقط هذه العناصر فى شريط الأدوات بالفورمة لتظهر تلقائيا الأيقونات التى تناظر هذه العناصر كما هو موضح بالشكل التالى :

شكل توضيحى :

Text Editor Tutorial
File Edit Help
New Open.. Save Exit Cut Copy Paste
RichEdit1
ActionManager1
ImageList1

لو أنك سحبت الأمر الخطأ واسقطه فى شريط الأدوات فى هذه الحالة تستطيع أن تسحبه إلى خارج شريط الأدوات مرة أخرى. كما تستطيع أيضا اختيار العنصر فى النافذة Object TreeView (كما هو موضح بالشكل التالى) ثم تضغط على المفتاح Delete بلوحة المفاتيح :



شكل توضيحى :



> **تستطيع أن تعيد ترتيب الأيقونات بشريط الأدوات بأن تسحب أى منهم إلى اليمين أو اليسار.**

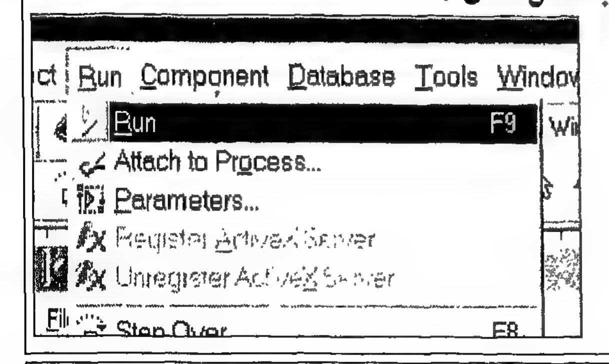
(٥) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها.

(٦) اضغط على المفتاح F9 بلوحة المفاتيح للبدء فى ترجمة وتشغيل المشروع ليظهر على الشاشة كما هو موضح بالشكل التالى :

شكل توضيحى :



تستطيع أيضا تشغيل المشروع بالنقر بالماوس على الأيقونة **Run** الموجودة بشريط الأدوات **Debug** أو بفتح القائمة **Run** واختيار **Run** منها كما هو موضح بالشكل التالى :





عندما تقوم بتشغيل المشروع الذى تعده فإن لغة **Delphi** تفتح البرنامج فى نافذة مرحلة التشغيل **Run-Time** وهذه النافذة تشبه الفورمة التى قمت بتصميمها كما هو موضح بالشكل التالى :

وستجد أن كل من القوائم وشريط الأدوات يعملان بالرغم من كون بعض الأوامر غير نشطة.



محـرر النصـوص الذى تعده أصبح لدية بالفعل العديد من الأدوات الوظيفية. وأنت تسـتطيع كـتابة مـا يحلـو لـك فـى مـنطقة النص كما تستطيع أيضا القيام بكل من عمليات القـص والنسـخ و اللصـق من خلال الأيقونات المخصصة لها بشريط الأدوات. ولكن على كل حال لايزال هناك الكثير الذى ينبغى عمله لتنشيط وتفعيل الأوامر الموجودة بالقوائم.

(٧) لكـى تعود مرة أخرى لمود التصميم Design-Time انقر بالماوس على الزر (×) الموجود بالركن الأيمن العلوى لنافذة التطبيق.

إخلاء منطقة النص من محتوياتها

عندما قمت بتشغيل البرنامج الذى تعده وجدت أن الاسم RichEdit1 قد ظهر فى مـنطقة النص. وأنت تستطيع إزالة هذا النص باستخدام Strings List Editor. ولو أنك لم تقم بتنظـيف الـنص الآن فـإن الـنص يـنـبغى مسـحه عـند الإعلان الابتدائى initializing الفورمة الأساسية فى آخر خطوة.

لتنظيف منطقة النص اتبع الخطوات التالية :

(١) فى الفورمة الأساسية انقر بالماوس على المكون RichEdit1.

(٢) فـى نـافذة Object Inspector وعـند الخاصـية Lines انقـر بالمـاوس عـلى الأيقونـة ⋯ المجـاورة للقـيمة TStrings ليظهر على الشاشة المحرر String List Editor الموضح فى الشكل التالى :

شكل توضيحى :

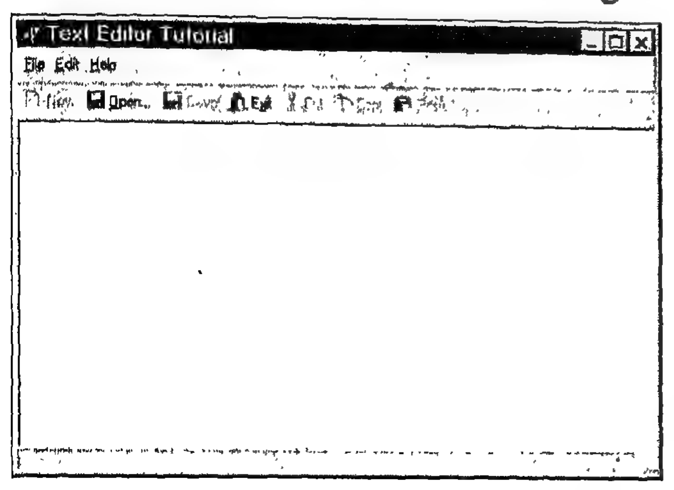
String List Editor
1 line
RichEdit1
Code Editor... | OK | Cancel | Help

(٣) اختر النص RichEdit1 والذى ترغب فى مسحه ثم قم بإزالته من المحرر Filter ثم اضغط على المفتاح Delete بلوحة المفاتيح ثم انقر بالماوس على المفتاح Ok.

(٤) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها ثم حاول تشغيل البرنامج مرة أخرى.

تلاحظ الآن أن منطقة تحرير النص أصبحت نظيفة تماما وذلك عند عرض الفورمة الأساسية على الشاشة كما هو موضح بالشكل التالى :

شكل توضيحى :



كتابة أدوات معاملة الأحداث Event handlers

حتى هذه المرحلة قمت بإعداد التطبيق الخاص بك بدون كتابة أى كود برمجى ولو حتى سطر واحد فقط. هذا وعن طريق استخدام Object Inspector لتحديد قيم الخصائص فى مرحلة التصميم Design-Time فإنك إذن قد استفدت تمام الاستفادة من بيئة التطوير RAD الخاصة بلغة Delphi.

فى هذا الجزء من الفصل سنقوم سويا بكتابة الإجراءات التى يطلق عليها أدوات معاملة الأحداث Event handlers والتى تستجيب لمدخلات المستخدم فى أثناء كون التطبيق فى مرحلة التشغيل Run-Time. وستقوم أيضا بإقامة الاتصال أو الربط بين أدوات



معاملـة الأحـداث Event handlers هذه والعناصر الموجودة في القوائم وشريط الأدوات ومن ثـم عـندما يـتم اختـيار أى عنصر يقوم التطبيق الذى تعده على الفور بتنفيذ الكود البرمجى الموجود بأداة معاملة الحدث الخاصة بهذا العنصر.

بالنسـبة للأفعـال الغـير قياسية ينبغى عليك إنشاء أداة معاملة خدث. أما بالنسبة للأفعال القياسية مثل الفعل الخاص بالأمر Exit بالقائمة File والفعل الخاص بالأمر Paste بالقائمـة Edit فـإن الأحـداث المرتـبطة بهـا تكـون داخـل الكود البرمجى الذى يتم تكوينه تلقائـيا لهـذه النوعـية مـن الأفعـال. ولكـن عـلى كـل حال يمكن القول بأنه بالنسبة لبعض الأفعال القياسية مثل الفعل الخاص بالأمر Save As بالقائمة File ستحتاج لأن تكتب أداة حدث خاصة بهذا الفعل (القياسى) وذلك من أجل تفصيل طريقة عمل هذا الأمر.

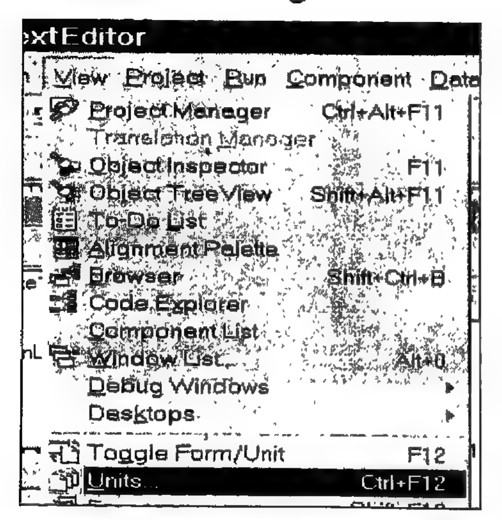


بسبب أن عناصر القوائم وشريط الأدوات كلها متركزة فى صندوق الحوار Action Manager لذلك تستطيع أن تنشأ أدوات معاملة الأحداث Event handlers من نفس صندوق الحوار هذا.

لكـى تنشأ أداة معاملة الحدث المرتبط بالأمر New (بالقائمة File) اتبع الخطوات التالية :

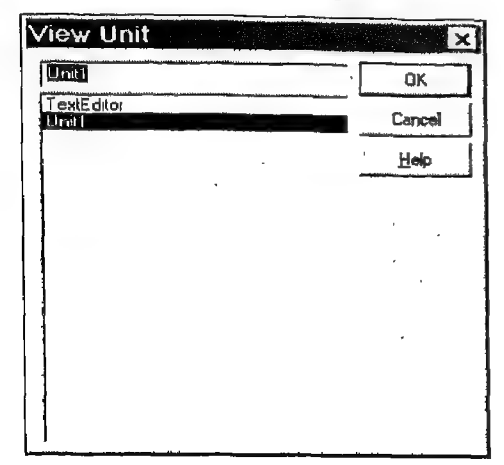
(١) افتح القائمة View واختر منها Units كما هو موضح بالشكل التالى :

شكل توضيحى :



ليظهر على الشاشة صندوق الحوار View Unit الموضح فى الشكل التالى :

شكل توضيحى :





يمكن فتح صندوق الحوار View Unit بطريقة أخرى وهى الضغط على المفتاحين Ctrl+F12 بلوحة المفاتيح.

بصندوق الحوار View Unit اختر الوحدة Unit1 ثم انقر بالماوس على المفتاح Ok وذلك لعرض الكود البرمجى الملحق بالفورمة Form1 كما هو موضح بالشكل التالى :

شكل توضيحى :

Unit1 pas

TForm1
Variables/Constants
Uses

Unit1

```
    FileExit1: TFileExit;
    HelpContents1: THelpContents;
    ImageList1: TImageList;
    ActionMainMenuBar1: TActionMainMenuBar;
    ActionToolBar1: TActionToolBar;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}
```

33: 1 Modified Insert Code Diagram

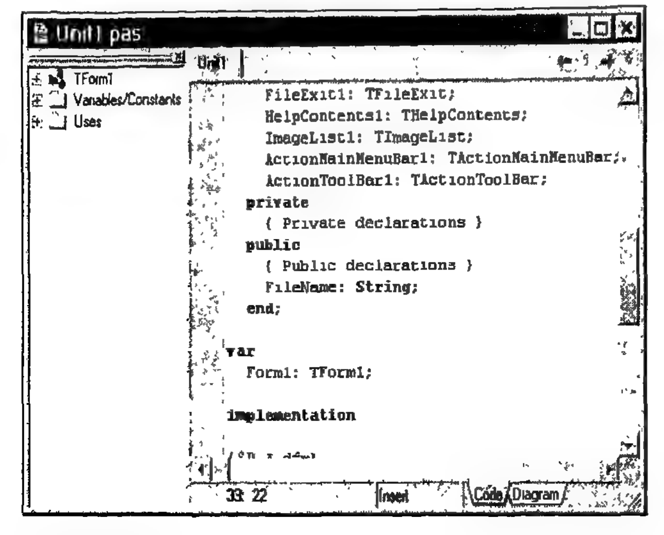
(٢) ستحتاج لأن تعلن عن اسم الملف الذى سيتم استخدامه فى أداة معاملة الحدث كما ستحتاج أيضا إلى إضافة خاصية مفصلة لاسم الملف هذا وذلك لجعل من

الممكن الوصول إليه من أى منطقة بالتطبيق. والآن وفى بداية ملف الوحدة Unit1.pas ابحث عن موضع المقطع الذى يضم مجموعة جمل الإعلانات العامة الخاصة بالقطاع Tform1 وفى السطر الذى يلى السطر {Public declarations} اكتب الآتى :

FileName: String;

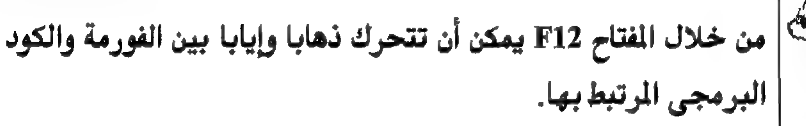
كما هو موضح بالشكل التالى :

شكل توضيحى :



(٣) اضغط على المفتاح F12 بلوحة المفاتيح للعودة مرة أخرى إلى الفورمة الأساسية.



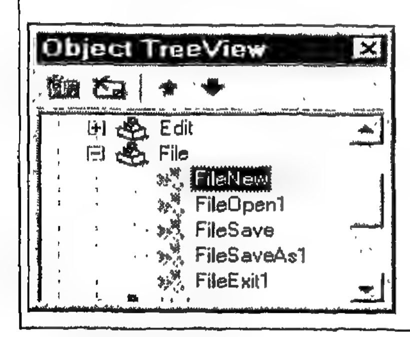

من خلال المفتاح F12 يمكن أن تتحرك ذهابا وإيابا بين الفورمة والكود البرمجى المرتبط بها.

(٤) انقر بالماوس نقرا مزدوجا على المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager.

(٥) فى صندوق الحوار Action Manager اختر القسم File بالقائمة Categories وبعد ذلك انقر بالماوس نقرا مزدوجا على الأيقونة New action.

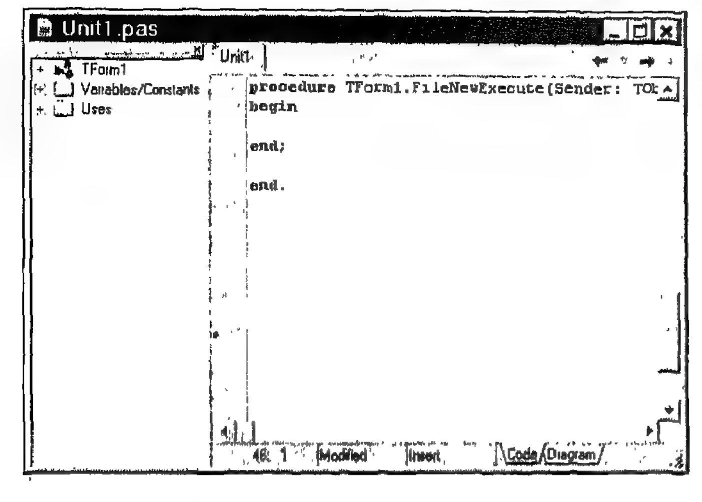


تستطيع أيضا أن تنقر بالماوس نقرا مزدوجا على الفعل FileNew الموجود بالقسم File وذلك فى النافذة Object TreeView كما هو موضح بالشكل التالى :



يـتم فتح محرر الكود البرمجى Code Editor وستجد أن مؤشر الكتابة داخل أداة معاملة الحدث الخاص بهذا الفعل كما هو موضح بالشكل التالى :

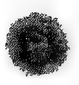
شكل توضيحى :



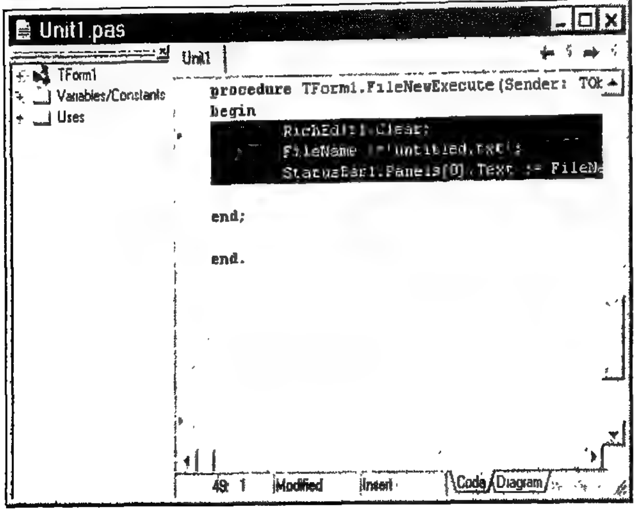
(٦) ضـع مؤشـر الكـتابة بمحـرر الكـود الـبرمجى (بـين Begin وEnd) ثـم اكتب السطور التالية :

```
RichEdit1.Clear;
FileName :='untitled.txt';
StatusBar1.Panels[0].Text := FileName;
```

ليصبح الكود البرمجى الخاص بهذا الحدث كما هو موضح بالشكل التالى :



شكل توضيحى :



(٧) افـتح القائمـة File ثـم اخـتر منها الأمر Save All ليتم حفظ التغييرات التى أجريتها الآن.



تستطيع تغيير حجم نافذة محرر الكود البرمجى لكى تقلل من الحاجة لاستخدام شريط الحركة الأفقى.

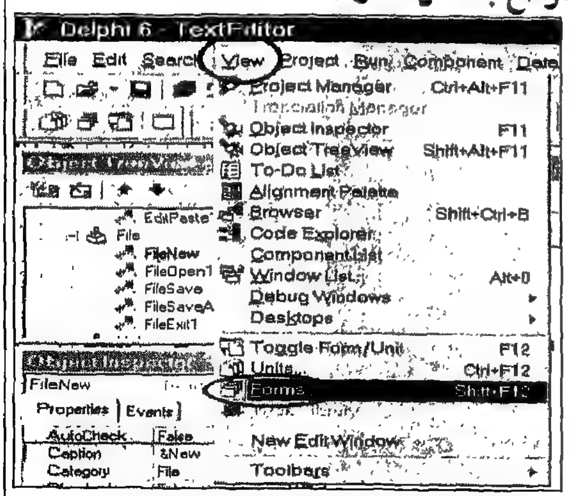
إنشاء أداة معاملة الحدث المرتبط بالأمر Open

لكـى يـتم فـتح ملـف فـى محـرر النصوص الذى تعده فإنك تحتاج لأن يظهر على الشاشة صندوق الحوار Open (وهو من ضمن صناديق الحوار القياسية ببيئة الويندوز). ولقد قمـت بـالفعل بإضافة الأمر القياسى Open إلى القائمة File وذلك بصندوق الحوار Action Manager والـذى يتضـمن تلقائـيا صـندوق الحـوار القياسـى Open. ولكـن عـلى كل حال سـتظل فـى حاجـة لتفصـيل أداة الحـدث المرتـبط بهـذا الأمر. وللقيام بذلك اتبع الخطوات التالية :

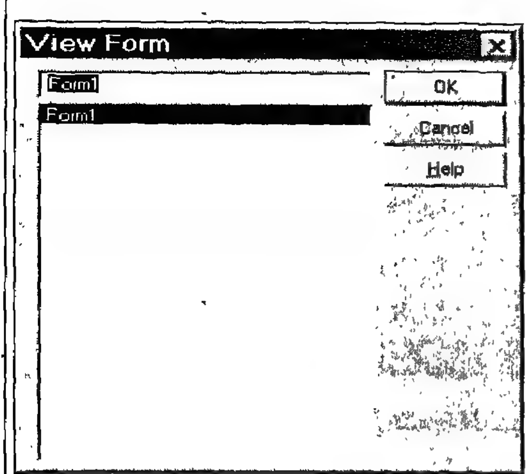
(١) اضـغط عـلى المفـتاح F12 بلوحـة المفاتـيح للذهـاب إلى الفورمة الأساسية بمود التصميم.



يمكن القيام بنفس المهمة عن طريق فتح القائمة View ثم اختيار الأمر Forms منها كما هو موضح بالشكل التالى :



ليظهر على الشاشة صندوق الحوار View form الموضح فى الشكل التالى :



ثم التعليم على Form1 ثم النقر بالماوس على المفتاح Ok.



يمكن أيضا فتح صندوق الحوار View Form عن طريق الضغط على المفتاحين Shift+F12.



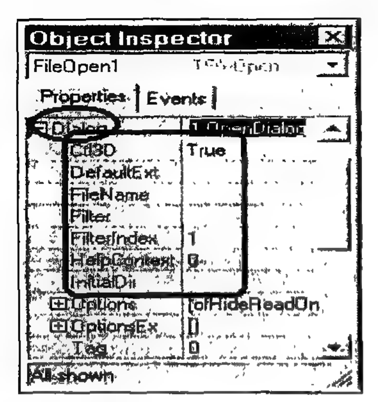
(٢) انقر بالماوس نقرا مزدوجا على أيقونة Action Manager لفتح صندوق الحوار Action Manager م اختر الفعل Open بالقسم File كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing Form1 ActionManager1
Toolbars Actions Options
(All Actions)
Categories: (No Category) Edit File Help (All Actions)
Actions: New Open... Ctrl+O Save Save As... Exit
Description
Opens an existing file.
To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.
Close

(٣) فى النافذة Object Inspector انقر بالماوس على العلامة (+) الموجودة على يسار الخاصية Dialog لعرض الخصائص الفرعية لهذه الخاصية الأساسية كما هو موضح بالشكل التالى :

شكل توضيحى :



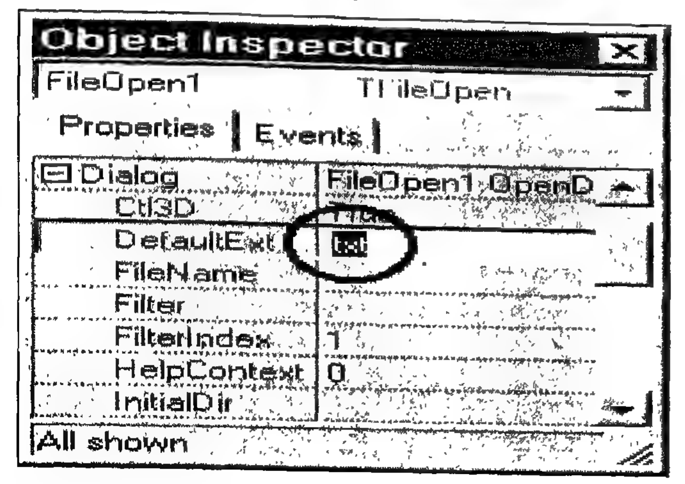


تقوم لغة Delphi بتسمية الخاصية OpenDialog للفعل FileOpen1 وهذه التسمية تعتبر تسمية افتراضية. وعندما يتم استدعاء الأمر Execute الخاص بالخاصية OpenDialog1 فإنه يتم فتح صندوق الحوار الأساسى Open لفتح الملفات.

(٤) قم بتحديد الخصائص التالية الخاصة بـFileOpen1.Dialog :

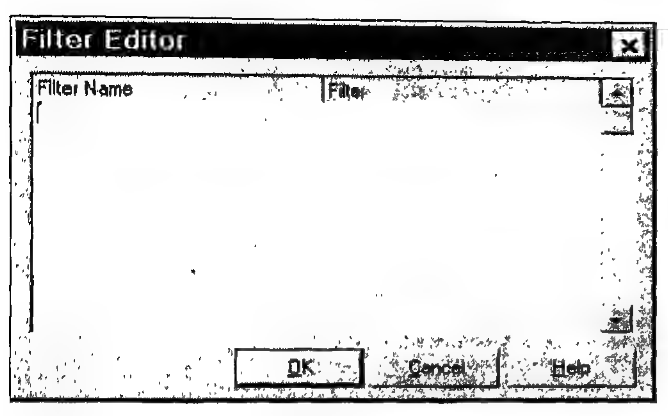
- عند الخاصية DefaultExt اكتب txt كما هو موضح بالشكل التالى :

شكل توضيحى :



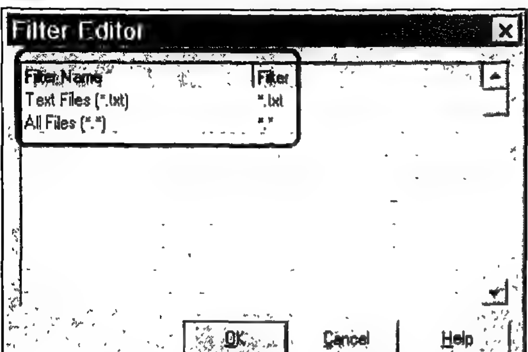
- انقر بالماوس نقرا مزدوجا على الحقل المجاور للخاصية Filter (او انقر بالماوس على الأيقونة ...) ليظهر على الشاشة صندوق الحوار Filter Editor الموضح فى الشكل التالى :

شكل توضيحى :



فى الخلية الموجودة بالصف الأول والعمود Filter Name اكتب Text files (*.txt) ثم فى الخلية الموجودة بالصف الأول والعمود Filter اكتب txt. ثم فى الخلية الموجودة بالصف الثانى والعمود Filter Name اكتب (*.*) All Filesثم فى الخلية الموجودة بالصف الثانى والعمود Filter اكتب *.* كما هو موضح بالشكل التالى :

شكل توضيحى :



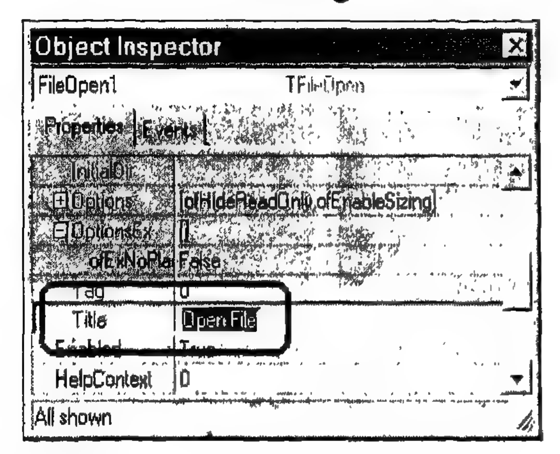
والآن انقر بالماوس على المفتاح OK لغلق صندوق الحوار. والآن ستجد القيمة المخصصة للخاصية Filter كما هو موضح بالشكل التالى :

شكل توضيحى :

Object Inspector
FileOpen1 TFileOpen
Properties | Events
Dialog | FileOpen1.OpenDialog
Ctl3D | True
DefaultExt | txt
FileName |
Filter | Text Files (*.txt)|*.txt|All Files (*.*)|*.*
FilterIndex | 1
HelpContext | 0
InitialDir |
All shown

• عند الخاصية Title اكتب Open File كما هو موضح بالشكل التالى :

شكل توضيحى :





كلمة Open File ستظهر كعنوان لصندوق الحوار القياسى Open.

(٥) انقر بالماوس على التبويب Events ليظهر على السطح بالنافذة Object Inspector كما هو موضح بالشكل التالى :

شكل توضيحى :

بهذا التبويب انقر بالماوس نقرا مزدوجا على الحدث OnAccept ليظهر على الشاشة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :

```
procedure TForm1.FileOpen1Accept(Sender: TC
begin

end;

end.
```

(٦) كما هو موضح بالشكل السابق تلاحظ أن مؤشر الكتابة موجود داخل أداة المعاملة لهذا الحدث.

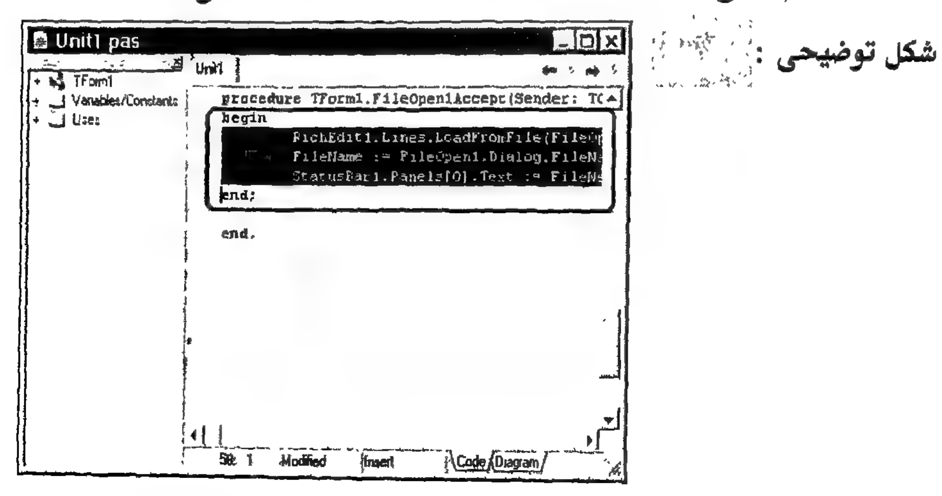


(٧) تحرك قليلا إلى جهة اليمين ثم اكتب مجموعة السطور التالية :

```
RichEdit1.Lines.LoadFromFile(FileOpen1.Dialog.FileName);
FileName := FileOpen1.Dialog.FileName;
StatusBar1.Panels[0].Text := FileName;
```

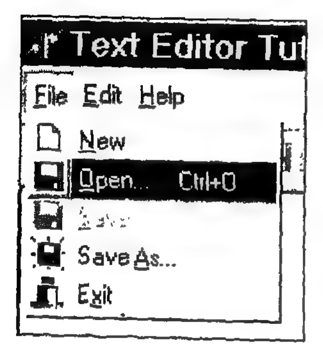
ومن ثم يصبح محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :



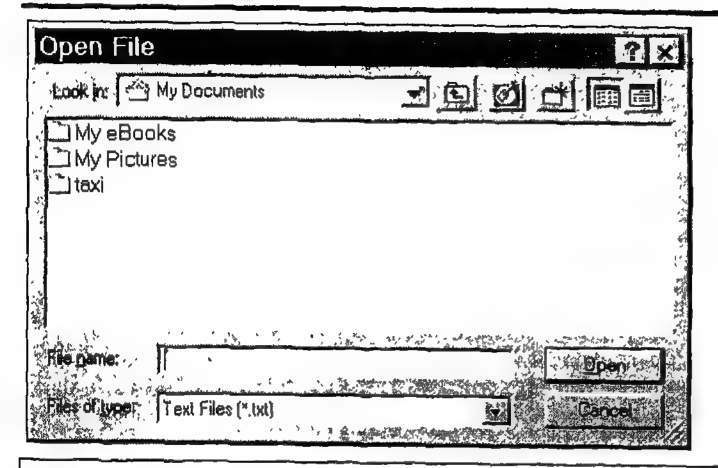
(٨) قـم الآن بتشـغيل التطبيق بالضغط على المفتاح F9 بلوحة المفاتيح ثم فى فورمة التطبيق الأساسية انقر بالماوس على الأيقونة Open [Open...] أو اضغط على المفـتاحين Ctrl+O بلوحـة المفاتـيح أو افـتح القائمـة File ثم اختر منها الأمر Open كما هو موضح بالشكل التالى :

شكل توضيحى :



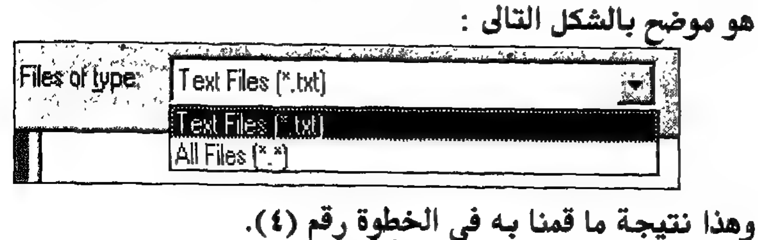
(٩) ليظهر على الشاشة صندوق الحوار Open File الموضح فى الشكل التالى :

شكل توضيحى :





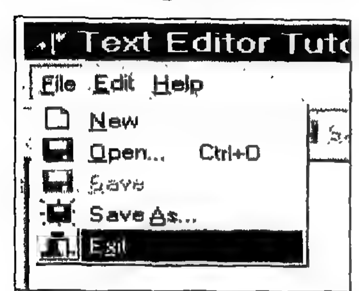
عندما تفتح القائمة المنسدلة Files of type: تلاحظ أنها تشتمل على كل من الاختيار (Text Files (*.txt والاختيار (*.*) All Files كما هو موضح بالشكل التالى :



وهذا نتيجة ما قمنا به فى الخطوة رقم (٤).

(١٠) انقر بالماوس على المفتاح Cancel لإغلاق صندوق الحوار Open File ثم افتح القائمة File واختر منها المر Exit كما هو موضح بالشكل التالى لإنهاء البرنامج :

شكل توضيحى :



(١١) افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التى أجريتها بالمشروع.

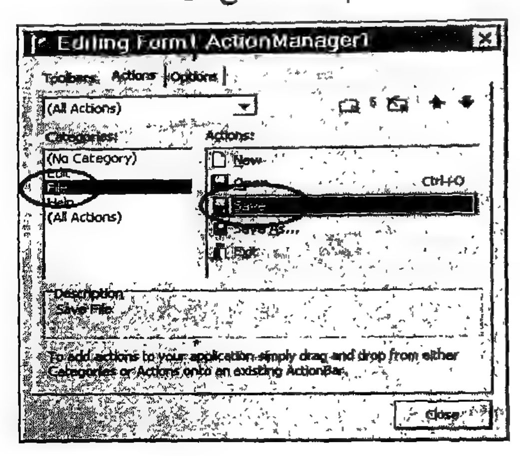
إنشاء أداة معاملة الحدث للأمر Save

لكى تنشأ أداة معاملة الحدث الخاص بالأمر Save اتبع الخطوات التالية :

(١) اضغط على المفتاح F12 بلوحة المفاتيح لعرض الفورمة على الشاشة. ثم انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا كان مغلقا).

(٢) انقر بالماوس نقرا مزدوجا على الفعل Save بالقسم File الموضح فى الشكل التالى :

شكل توضيحى :



ليتم فتح محرر الكود البرمجى Code Editor وستجد أن مؤشر الكتابة موجود داخل أداة معاملة الحدث الخاص بالفعل Save كما هو موضح بالشكل التالى :

شكل توضيحى :

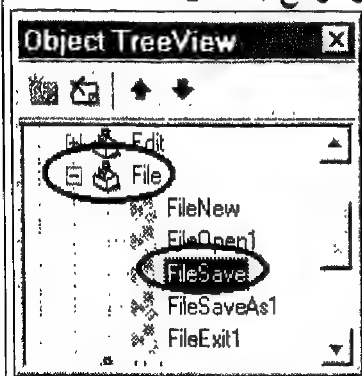
```
procedure TForm1.FileSaveExecute(Sender: TC
begin

end;

end.
```



تستطيع أيضا النقر بالماوس نقرا مزدوجا على الفعل FileSave بالقسم File بنافذة Object TreeView كما هو موضح بالشكل التالى :

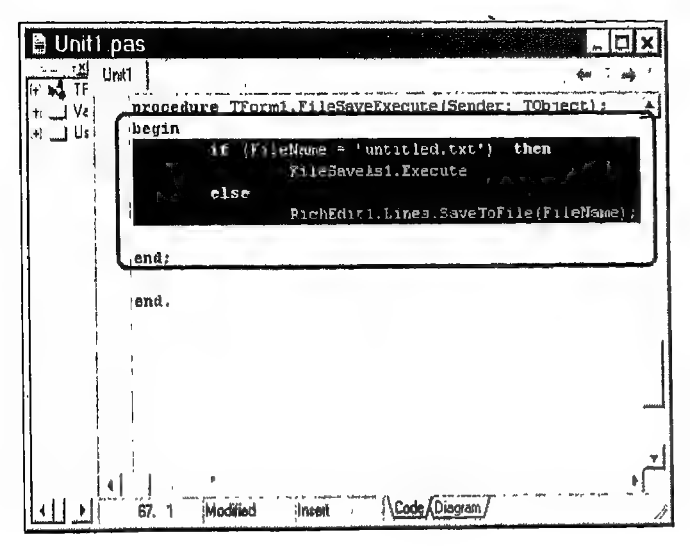


(٣) تحـرك بمؤشـر الكـتابة إلى الـيمين قلـيلا بالسطر الفارغ بين Begin وEnd ثم اكتب السطور التالية :

```
if (FileName = 'untitled.txt')  then
        FileSaveAs1.Execute
else
RichEdit1.Lines.SaveToFile(FileName);
```

كما هو موضح بالشكل التالى :

شكل توضيحى :







هذا الكود البرمجي يخبر محرر النصوص بأن يعرض صندوق الحوار Save As إذا كان الملف المطلوب حفظه لم يتم تسميته بعد ومن ثم يستطيع المستخدم تخصيص اسم له من خلال صندوق الحوار Save As. ولكن في حالة وجود اسم للملف فإنه يتم حفظه بنفس هذا الاسم.



الخاصية BeforeExecute للفعل FileSaveAs1 تقوم تلقائيا بتكوين اسم للأمر Save As وهذا لا يحدث بالنسبة للأمر Save بالقائمة File.

والآن افتح القائمة File واختر منها الأمر Save All لحفظ التغييرات التي أجريتها بالمشروع.

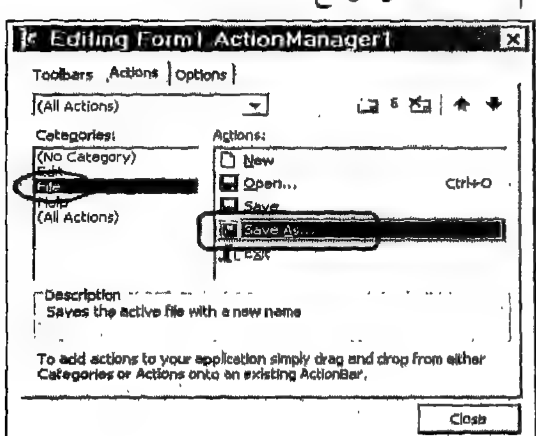
إنشاء أداة معاملة الحدث المرتبط للأمر Save As

عندما يتم استدعاء الأسلوب Execute الخاص بـ SaveDialog يظهر على الشاشة صندوق الحوار القياسي Save As لحفظ الملفات. هذا ولكي تنشأ أداة معاملة الحدث المرتبط للأمر Save As اتبع الخطوات التالية :

(١) اضغط على المفتاح F12 بلوحة المفاتيح لعرض الفورمة على الشاشة. ثم انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا كان مغلقا).

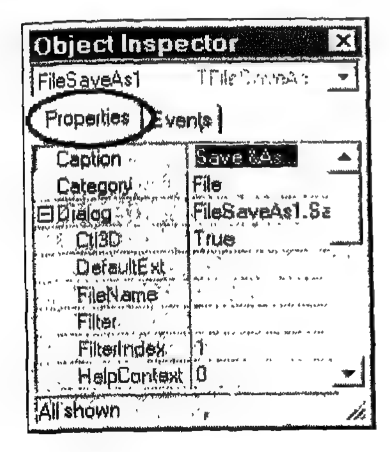
(٢) اختر الفعل SaveAs بالقسم File كما هو موضح بالشكل التالي :

شكل توضيحي :



(٣) فى النافذة Object Inspector انقر بالماوس على التبويب Properties كما هو موضح بالشكل التالى :

شكل توضيحى :



قم بتحديد قيم الخصائص التالية لصندوق الحوار FileSaveAs1 (الذى تخصص له لغة Delphi الاسم الافتراضى FileSaveAs1).

(٤) انقر بالماوس على العلامة (+) الموجودة على يسار الخاصية Dialog ثم قم بتحديد قيم الخصائص التالية :

- عند الخاصية DefaultExt اكتب txt كما هو موضح بالشكل التالى :

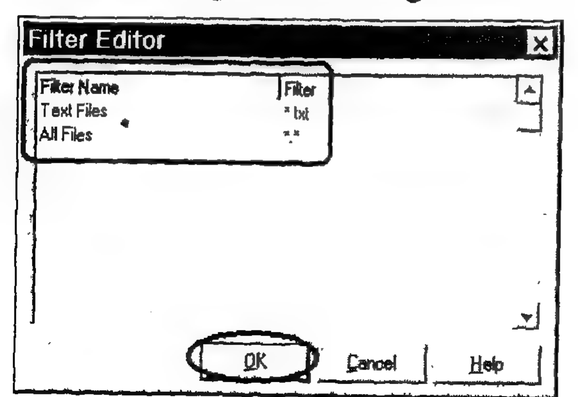
شكل توضيحى :

- انقر بالماوس نقرا مزدوجا على الحقل المجاور للخاصية Filter لعرض صندوق الحوار Filter Editor والذى يتم من خلاله توصيف الفلاتر الخاصة بأنواع الملفات



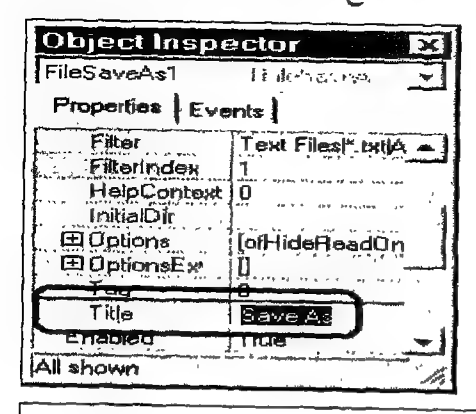
مثلما حدث بالنسبة لصندوق الحوار Open. والآن وفى الصف الأول وبالعمود Filter Name اكتب (Text Files (*.txt ثم فى العمود Filter اكتب *.txt. أما فى الصف الثانى وبالعمود Filter Name اكتب (*.*) All Files ثم فى العمود Filter اكتب *.* ثم انقر بالماوس على المفتاح Ok كما هو موضح بالشكل التالى :

شكل توضيحى :



- عند الخاصية Title اكتب Save As كما هو موضح بالشكل التالى :

شكل توضيحى :



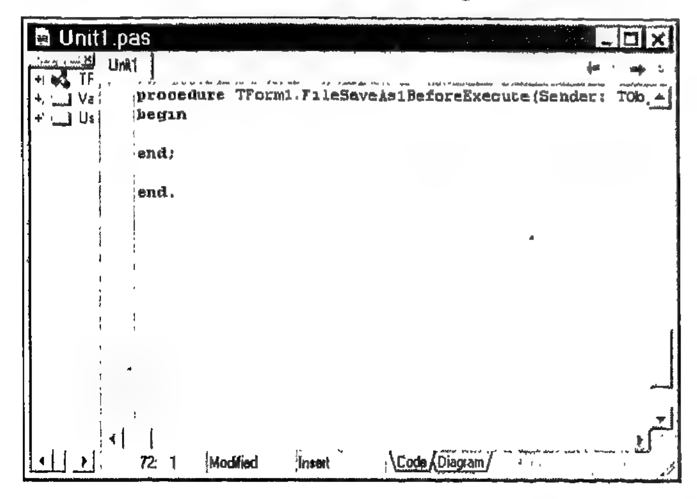


القيمة التى يتم تخصيصها للخاصية Title تظهر بشريط العنوان لصندوق الحوار Save As كما سنرى بعد ذلك.

(٥) فى النافذة Object Inspector انقر بالماوس على التبويب Events ليظهر على السطح وفيه انقر بالماوس نقرا مزدوجا على الحقل المجاور للخاصية BeforeExecute ليظهر هيكل الكود البرمجى الخاص بـ

FileSaveAs1BeforeExecute وذلك فى محرر الكود البرمجى Code Editor والذى فيه نشاهد مؤشر الكتابة داخل الكود البرمجى لأداة معاملة الحدث للأمر SaveAs كما هو موضح بالشكل التالى :

شكل توضيحى :

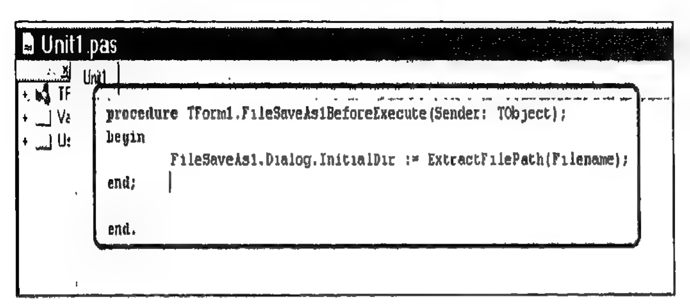


(٦) بالسطر الموجود به مؤشر الكتابة بمحرر الكود البرمجى اكتب الكود الآتى :

```
FileSaveAs1.Dialog.InitialDir := ExtractFilePath(Filename);
```

كما هو موضح بالشكل التالى :

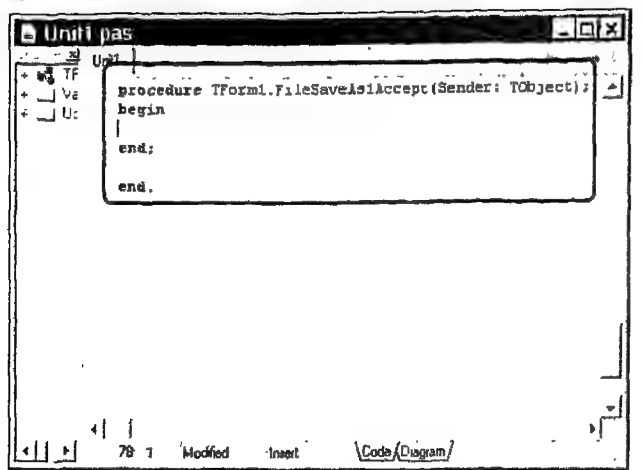
شكل توضيحى :



(٧) فى النافذة Object Inspector لابد أن يكون التبويب Events لا يزال موجود على السطح. وفيه انقر بالماوس نقرا مزدوجا على الحقل المجاور للحدث OnAccept لكى يظهر هيكل الكود البرمجى الخاص بـ FileSaveAs1Accept وذلك فى محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :



شكل توضيحى :



(٨) بالسطر الموجود به مؤشر الكتابة اكتب الكود التالى :

```
RichEdit1.Lines.SaveToFile(FileSaveAs1.Dialog.FileName);
FileName := FileSaveAs1.Dialog.FileName;
StatusBar1.Panels[0].Text := FileName;
```

كما هو موضح بالشكل التالى :

شكل توضيحى :

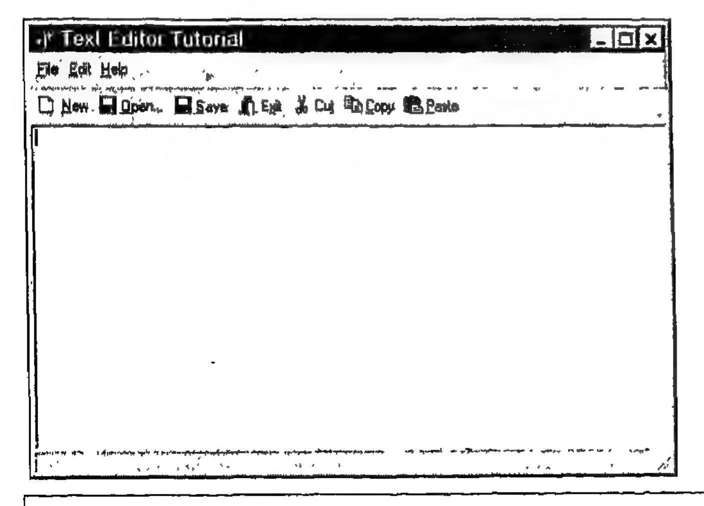
```
Unit1.pas
procedure TForm1.FileSaveAs1Accept(Sender: TObject);
begin
    RichEdit1.Lines.SaveToFile(FileSaveAs1.Dialog
    FileName := FileSaveAs1.Dialog.FileName;
    StatusBar1.Panels[0].Text := FileName;
end;

end.
```

(٩) افتح القائمة File واختر منها الأمر Save All لحفظ كافة التغييرات التى أجريتها بالمشروع.

(١٠) لكى تشاهد الحالة التى وصل إليها التطبيق قم بتشغيله بالضغط على المفتاح F9 بلوحة المفاتيح لتظهر على الشاشة الفورمة الأساسية لهذا التطبيق كما هو موضح بالشكل التالى :

شكل توضيحى :





تلاحظ أن أغلب الأوامر الموجودة بالقوائم وكذلك أغلب الأيقونات الموجودة بشريط الأدوات أصبحت تعمل الآن بالرغم من إنك لم تنتهى بعد من المشروع.

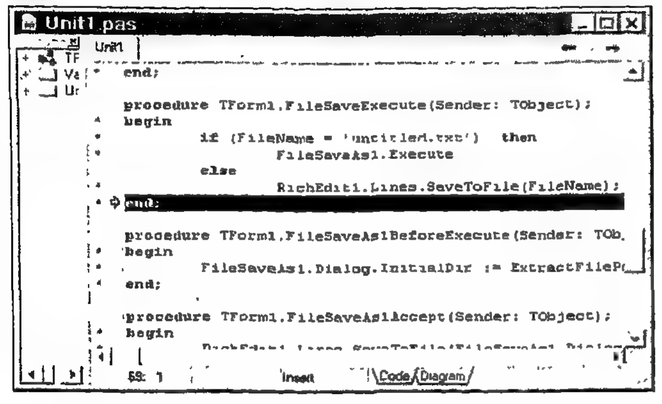
لو أنك تشاهد أى رسائل خطأ عند النقر بالماوس على أى أيقونة من أيقونات المشروع مثل رسالة الخطأ الموضحة فى الشكل التالى :

شكل توضيحى :

Debugger Exception Notification
Project TextEditor.exe raised exception class EFCreateError with message 'Cannot create file'. Process stopped. Use Step or Run to continue.
OK Help
View CPU Window

فى هذه الحالة انقر بالماوس على المفتاح Ok لكى تذهب إلى موضع الخطأ فى الكود البرمجى بنافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

شكل توضيحى :



وبهـذه الطـريقة تسـتطيع إصلاح الخطأ الذى حدث بالكود البرمجى. وعلى العموم عليك أن تتأكد من إنك اتبعت بطريقة صحيحة الخطوات التى ذكرناها قبل ذلك.

(١١) للعـودة مـرة أخـرى إلى مود التصميم انقر بالماوس على الزر (×) بالركن الأيمن العلوى لنافذة التطبيق.

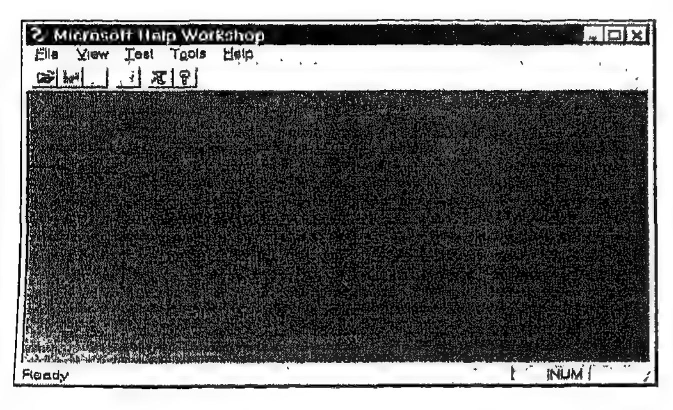
(١٢) افـتح القائمـة File واخـتر مـنها الأمـر Save All (او اضـغط على مجموعة المفاتيح Shift+Ctrl+S بلوحة المفاتيح) لحفظ التغييرات التى أجريتها بالمشروع.

إنـشاء ملف مساعدة

إنهـا لفكـرة جيدة أن يتم إنشاء ملف مساعدة يشرح كيفية استخدام التطبيق الذى تعده. وفـى هـذا الصـدد نقـول إن لغـة Delphi تعمـل عـلى توفـير ورشـة العمـل Microsoft Help Workshop والموجـودة فـى المسـار C:\Program Files\Borland\Delphi6\Help\Tools والموضحة فى الشكل التالى :



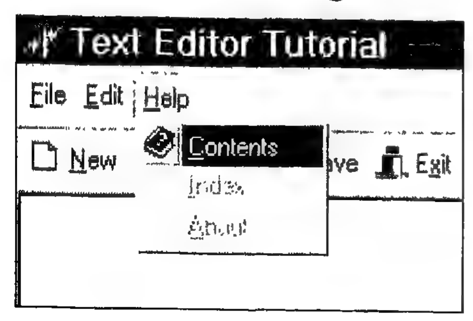
شكل توضيحى :



وورشة العمل هذه تتضمن معلومات عن تصميم وترجمة compiling أى ملف من ملفات المساعدة التى تعمل ببيئة الويندوز.

هذا وفى تطبيق محرر النصوص الذى نحن بصدده هنا نود أن نجعل المستخدمين له لديهم القدرة على الوصول إلى ملف المساعدة (المشتمل على التبويب Contents والتبويب Index) عن طريق الأمر Contents بقائمة Help كما هو موضح بالشكل التالى :

شكل توضيحى :



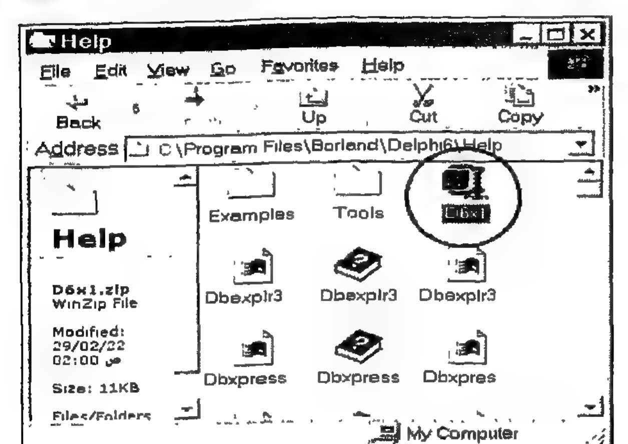
فيما سبق قمت بإنشاء كل من الفعل HelpContents والفعل HelpIndex فى صندوق الحوار Action Manager وذلك بهدف عرض التبويب Contents أو التبويب Index بملف المساعدة المترجم. والآن أنت فى حاجة لأن تخصص قيم ثابتة لمعاملات المساعدة كما تحتاج أيضا لإنشاء أدوات معاملة الأحداث التى تعمل على عرض ما يرغبه المستخدم من الموضوعات الموجودة بنظام المساعدة.

لكى تستخدم أوامر المساعدة عليك أن تقوم بإنشاء وترجمة ملف مساعدة يعمل ببيئة الويندوز. ونود هنا القول بأن إنشاء ملفات المساعدة موضوع كبير إلى حد ما ولا يمكننا تناوله بالتفصيل فى هذا الفصل. ولكن على العموم تستطيع تحميل كل من ملف النموذج المسمى TextEditor.rtf وملف المساعدة المسمى TextEditor.hlp وملف المحتويات المسمى TextEditor.cnt وذلك عن طريق الخطوات التالية :

(١) من خلال المسار C:\Project Files\Borland\Delphi6\Help افتح الملف D6X1.zip الموضح فى الشكل التالى :



شكل توضيحى :



(٢) يظهـر الآن عـلى الشاشـة صندوق الحوار الافتتاحى للبرنامج WinZip كما هو موضح بالشكل التالى :

شكل توضيحى :

WinZip

THE ARCHIVE UTILITY FOR WINDOWS

THANK YOU FOR TRYING WINZIP!

This is a fully functional unregistered version for evaluation use only.
The registered version does not display this notice.

You can order the registered version online, by phone, or by mail.
Immediate online delivery is available from www.winzip.com.

View Evaluation License | Enter Registration Code...

I understand that I may use WinZip only for evaluation purposes, subject to the terms of the Evaluation License, and that any other use requires payment of the registration fee.

Quit | I Agree | Ordering Info

(٣) انقـر بالماوس على المفتاح I Agree ليتم فتح نافذة البرنامج وبها نشاهد محتويات الملف D6X1.zip كما هو موضح بالشكل التالى :

شكل توضيحى :

WinZip (Unregistered) - D6X1.ZIP

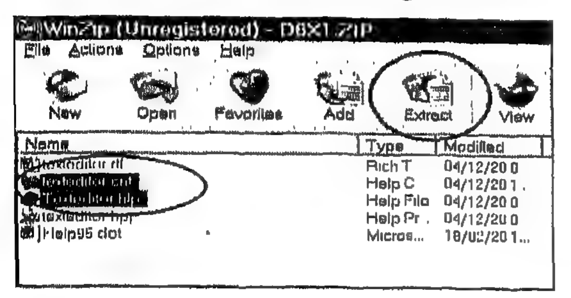
File Actions Options Help

New Open Favorites Add Extract View CheckOut Wizard

Name	Type	Modified	Size	R	Pack
texteditor.rtf	Rich T	04/12/20 0	26,295	84%	4,139
texteditor.cnt	Help C	04/12/20 1	242	45%	133
Texteditor.hlp	Help File	04/12/20 0	16,384	89%	1,741
texteditor.hpj	Help Fr	04/12/20 0	489	35%	319
Helps5.dot	Macros	18/02/20 1	18,944	79%	3,949

(٤) علم بالماوس على كل الملفات التى لها الامتداد hlp. والامتداد cnt. ثم انقر بالماوس على المفتاح Extract كما هو موضح بالشكل التالى :

شكل توضيحى :



(٥) يظهر على الشاشة صندوق الحوار Extract الموضح فى الشكل التالى :

شكل توضيحى :

Extract
Extract to
Folders/drives:
Files
Selected files
All files
Files:
Overwrite existing files
Skip older files
Use folder names
Desktop
My Computer
3 5 Floppy (A:)
Sherif (C:)
Sherif (D:)
Sherif (E:)
(F:)
(G:)
Network Neighborhood
My Documents
My Briefcase
Extract
Cancel
Help
New Folder...

(٦) بقائمة العرض Folders/drives ابحث عن المجلد TextEditor الذى أنشأته فى بداية هذا الفصل لتخزين ملف المشروع الذى تعده به. وهذا المجلد موجود بالمسار C:\Project Files\Borland\Delphi6\Projects\TextEditor كما هو موضح بالشكل التالى :

شكل توضيحى :

Extract
Extract to:
C:\Program Files\Borland
Folders/drives:
Files
Selected files
All files
Files:
Overwrite existing files
Skip older files
Use folder names
Imports
Lib
MergeModules
Objrepos
Ocx
Projects
Bpl
TextEditor
Source
Chat
Extract
Cancel
Help
New Folder...



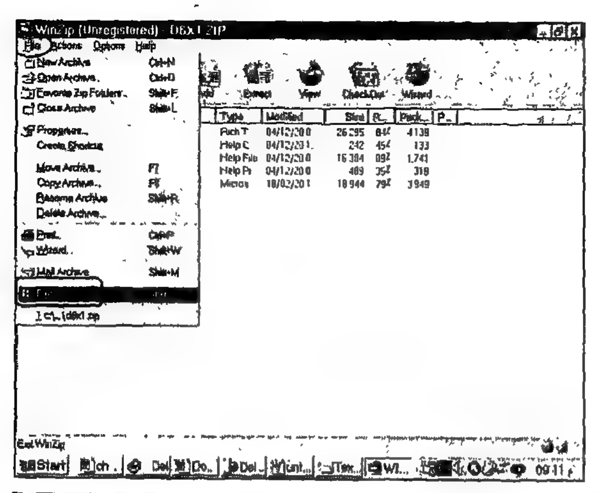
(٧) انقر بالماوس على المفتاح Extract ليتم استخلاص الملفات المختارة ووضعها فى المجلد المختار كما هو موضح بالشكل التالى :

شكل توضيحى :

(٨) لقد اختفى صندوق الحوار Extract من على الشاشة وعدت مرة أخرى إلى نافذة البرنامج WinZip.

(٩) قم بغلق البرنامج WinZip بفتح القائمة File واختيار الأمر Exit منها كما هو موضح بالشكل التالى :

شكل توضيحى :





تستطيع أن تستخدم أى ملف مساعدة له امتداد HLP أو CNT (مثل ملفات المساعدة HLPالتى تأتى مع لغة Delphi والملفات CNT الملحقة

بها) بالمشروع الذى تعده. وفى هذه الحالة ينبغي عليك تغيير اسماء هذه الملفات لتصب TextEditor.hlp وTextEditor.cnt وذلك لكى يتمكن التطبيق الذى تعده من العثور على هذه الملفات.

إنشاء أداة معاملة الحدث الخاص بالأمر Help Contents

لإنشـاء أداة معاملـة الحـدث الخـاص بالأمـر Contents بالقائمـة Help اتـبع الخطوات التالية :

(١) انقـر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوحا بالفعل).

(٢) فـى صـندوق الحـوار Action Manager اخـتر Help مـن القائمـة المنسـدلة Categories ثـم انقـر بالمـاوس نقـرا مـزدوجا عـلى الفعـل Contents بقائمـة العرض Actions كما هو موضح بالشكل التالى :

شكل توضيحى :

Editing Form1.ActionManager1

Toolbars | Actions | Options

(All Actions)

Categories: (No Category), Edit, File, Help, (All Actions)

Actions: Index, About, Contents

Description: Help Contents

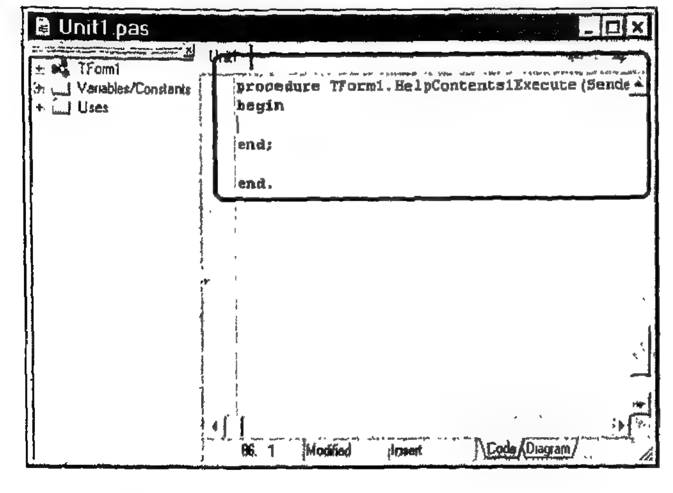
To add actions to your application simply drag and drop from either Categories or Actions onto an existing ActionBar.

Close

الآن يظهـر عـلى الشاشـة محـرر الكـود البرمجى Code Editor وبه نشاهد مؤشر الكـتابة داخـل الهـيكل الـبرمجى لأداة معاملـة الحـدث الخـاص بالأمر Contents كما هو موضح بالشكل التالى :



شكل توضيحى :



(٣) بالسطر الذى يسبق الكلمة begin مباشرة اكتب الكود التالى :

```
const
HELP_TAB = 15;
CONTENTS_ACTIVE = -3;
```

كما هو موضح بالشكل التالى :

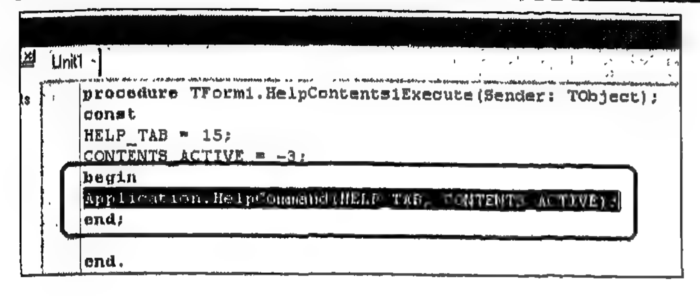
شكل توضيحى :

ثم بالسطر الذى يلى الكلمة Begin مباشرة اكتب الكود التالى :

```
Application.HelpCommand(HELP_TAB, CONTENTS_ACTIVE);
```

كما هو موضح بالشكل التالى :

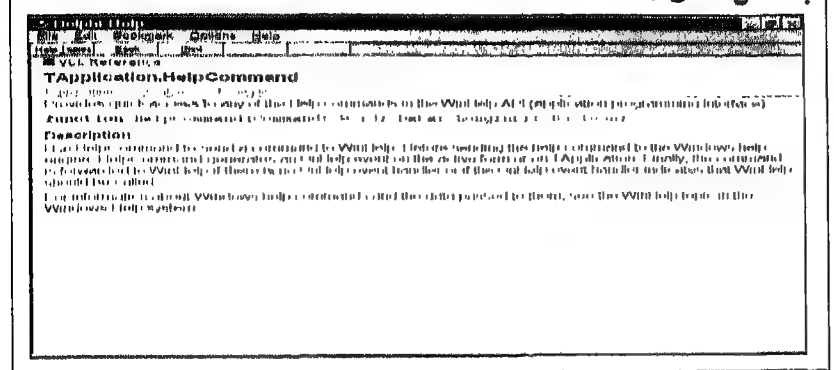
شكل توضيحى :



هـذا الكـود يعمـل عـلى تخصـيص القـيم الثابـتة إلى المعـاملات الخاصـة بالأمـر HelpCommand. ونـود هـنا القول بأن تخصيص ١٥ إلى المعامل HELP_TAB يؤدى إلى عـــرض صـــندوق الحـــوار Help كمـــا أن تخصـــيص القـــيمة –٣ إلى المعـــامل CONTENTS_ACTIVE يؤدى إلى عرض التبويب Contents بصندوق الحوار Help.



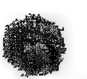
إذا أردت على الحصول على المساعدة حول الحدث HelpCommand ضع مؤشر الكتابة بجوار HelpCommand فى محرر الكود البرمجى Code Editor ثم اضغط على المفتاح F1 بلوحة المفاتيح ليظهر على الشاشة صندوق الحوار Help وبه الموضوع الذى ترغبه كما هو موضح بالشكل التالى :



إنشاء أداة معاملة الحدث للأمر Help Index

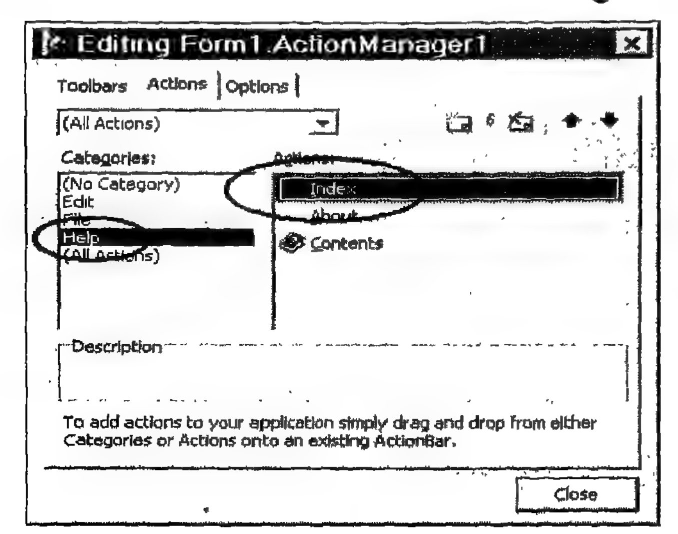
لإنشاء أداة معاملة الحدث للأمر Help Index اتبع الخطوات التالية :

(١) انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager بالفورمة لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوحا بالفعل).



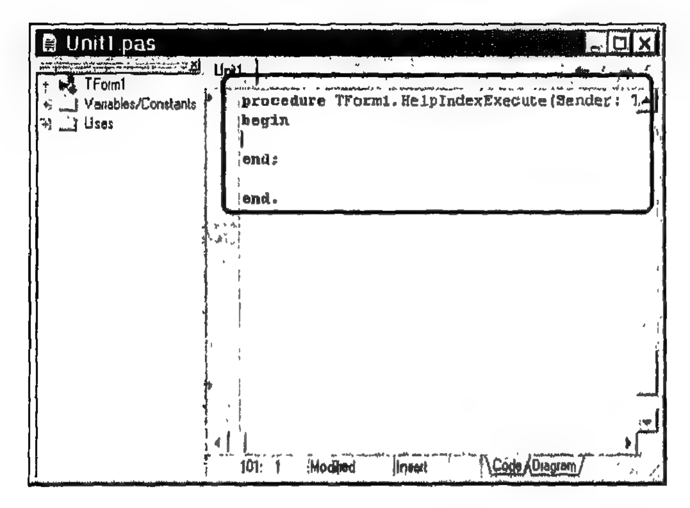
(٢) فـي صـندوق الحـوار Action Manager اخـتر Help مـن القائمـة المنسـدلة Categories ثـم انقـر بالماوس نقرا مزدوجا على الفعل Index بقائمة العرض Actions كما هو موضح بالشكل التالى :

شكل توضيحى :



الآن يظهـر عـلى الشاشـة محـرر الكـود البرمجى Code Editor وبه نشاهد مؤشر الكـتابة داخـل الهـيكل البرمجى لأداة معاملة الحدث الخاص بالأمر Index كما هو موضح بالشكل التالى :

شكل توضيحى :

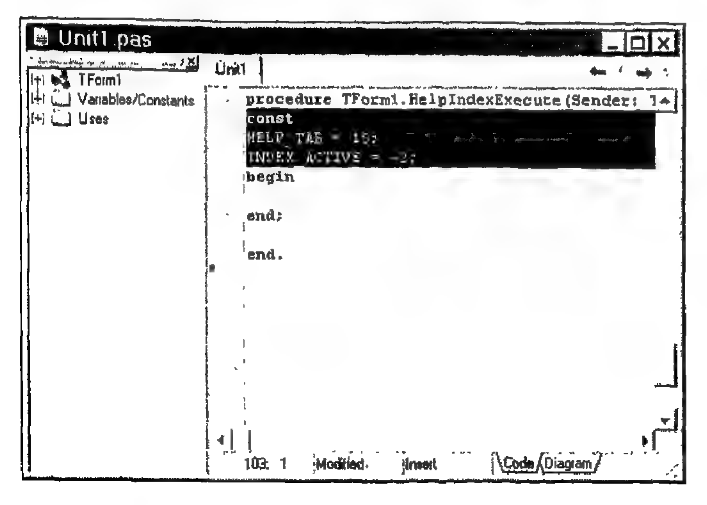


(٤) بالسطر الذى يسبق الكلمة begin مباشرة اكتب الكود التالى :

```
const
HELP_TAB = 15;
INDEX_ACTIVE = -2;
```

كما هو موضح بالشكل التالى :

شكل توضيحى :



ثم بالسطر الذى يلى الكلمة Begin مباشرة اكتب الكود التالى :

```
Application.HelpCommand(HELP_TAB, INDEX_ACTIVE);
```

كما هو موضح بالشكل التالى :

شكل توضيحى :

```
Unit1.pas
procedure TForm1.HelpIndexExecute(Sender: TObject);
const
HELP_TAB = 15;
INDEX_ACTIVE = -2;
begin
Application.HelpCommand(HELP_TAB, INDEX_ACTIVE);
end;

end.
104: 49   Modified   Insert   Code/Diagram
```



هذا الكود يعمل على تخصيص القيم الثابتة إلى المعاملات الخاصة بالأمر HelpCommand. ونود هنا القول بأن تخصيص ١٥ إلى المعامل HELP_TAB يؤدى إلى عرض صندوق الحوار Help كما أن تخصيص القيمة –٢ إلى المعامل CONTENTS_ACTIVE يؤدى إلى عرض التبويب Index بصندوق الحوار Help.

إنشاء صندوق الحوار About

العديد من التطبيقات تكون مشتملة على صندوق الحوار About الذى يعرض معلومات عن المنتج مثل اسمه ورقم الإصدار والعلامات التجارية logos كما قد يشتمل على معلومات عن الاستخدام القانونى للمنتج وحقوق الملكية.

لقد إنتهيت حتى الآن من إضافة الأمر About للقسم Help بصندوق الحوار Action Manager.

لإضافة صندوق الحوار About اتبع الخطوات التالية :

(١) افتح القائمة File ثم اختر New ثم اختر Other كما هو موضح بالشكل التالى:

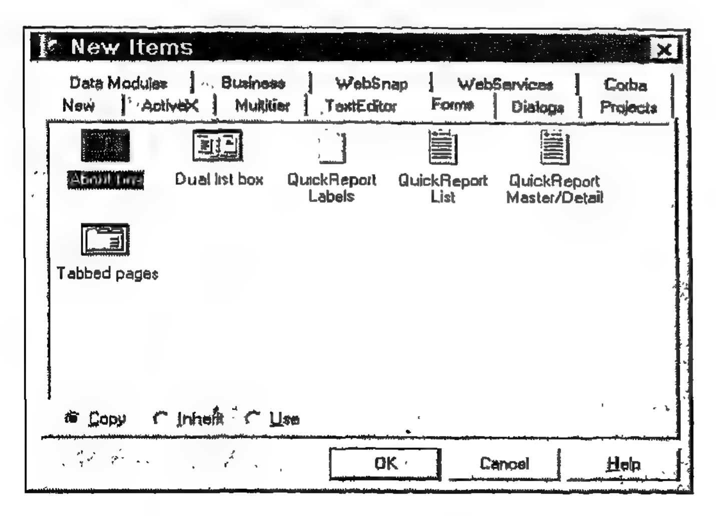
شكل توضيحى :

ليظهر على الشاشة صندوق الحوار New Items الموضح فى الشكل التالى :

شكل توضيحى :

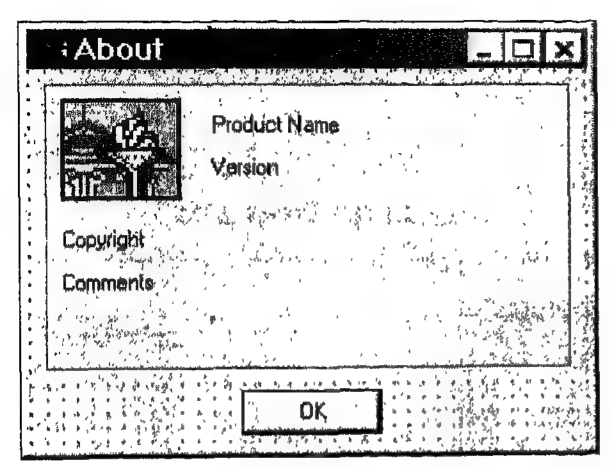
انقـر بالمـاوس عـلى التبويـب Forms لـيظهر عـلى السـطح كما هو موضح بالشكل التالـى :

شكل توضيحى :



(٢) بالتبويـب Forms انقـر بالمـاوس نقرا مزدوجا على الأيقونة About Box ليتم إنشـاء فورمـة جديـدة تعمـل عـلى تبسيط عملية إنشاء صندوق الحوار About والشكل التالى يوضح لنا هذه الفورمة الجديدة :

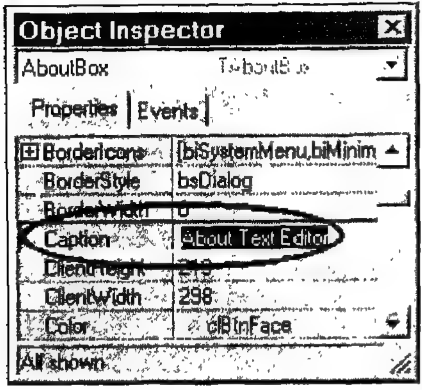
شكل توضيحى :



(٣) اخـتر الفورمـة نفسـها (بالـنقر بالمـاوس عـلى الجـزء المنقط بها) ثم فى النافذة Object Inspector اجعـل قيمة الخاصية Caption عبارة عن About Text Editor كما هو موضح بالشكل التالى :



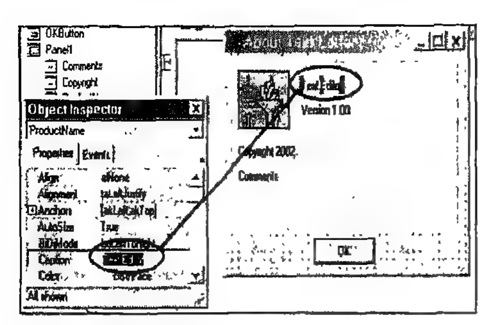
شكل توضيحي :



(٤) فـى هذه الخطوة ستقوم بتغيير النصوص (العناوين Labels) الموجودة بالفورمة About وذلك كالآتى :

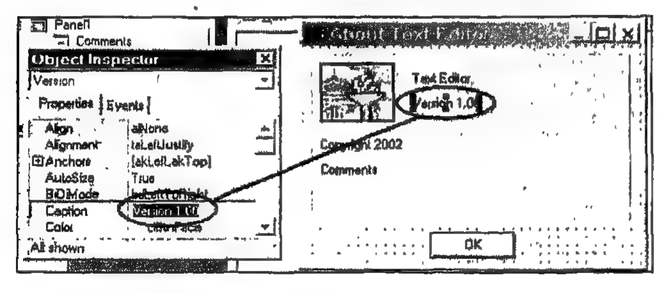
- انقـر بالمـاوس عـلى Product name بالفورمـة About ثم فـى الـنافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Text Editor كما هو موضح بالشكل التالى :

شكل توضيحي :



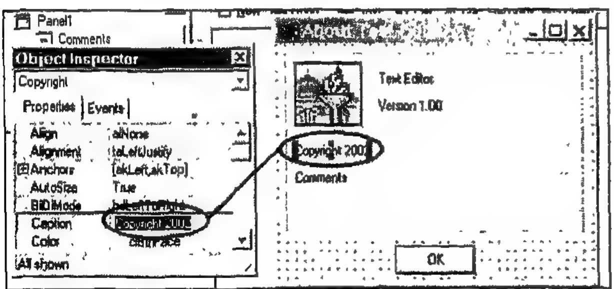
- انقر بالماوس على Version بالفورمة About ثم فى النافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Version 1.00 كما هو موضح بالشكل التالى :

شكل توضيحي :



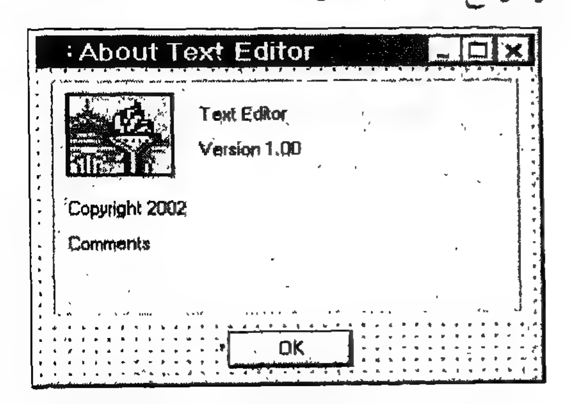
- انقر بالماوس على Copyright بالفورمة About ثم فى النافذة Object Inspector اجعل قيمة الخاصية Caption عبارة عن Copyright 2002 كما هو موضح بالشكل التالى :

شكل توضيحى :



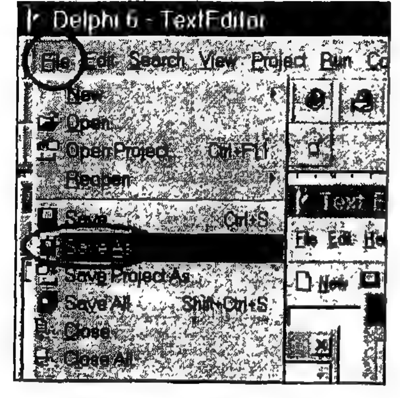
والآن تصبح الفورمة About كما هو موضح بالشكل التالى :

شكل توضيحى :



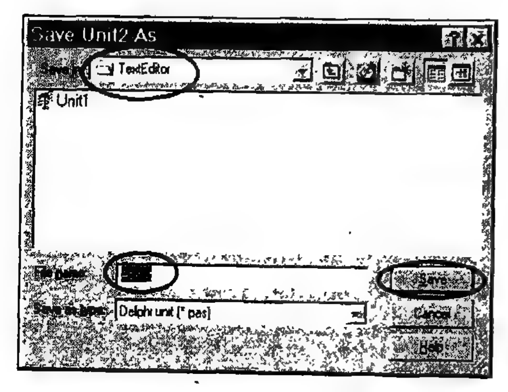
(٥) احفظ فورمة صندوق الحوار About بفتح القائمة File ثم اختيار الأمر Save As منها كما هو موضح بالشكل التالى :

شكل توضيحى :



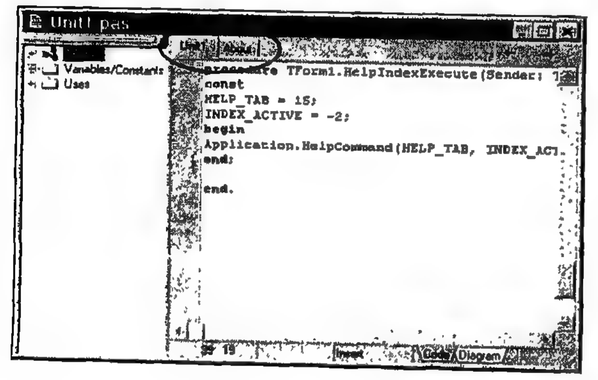
ليظهر على الشاشة صندوق الحوار Save Unit2 As ومن خلاله احفظ هذه الفورمة باسم About.pas كما هو موضح بالشكل التالى :

شكل توضيحى :



(٦) فى محرر الكود البرمجى Code Editor بلغة Delphi ينبغى أن يكون به تبويب للوحدة Unit1 وتبويب آخر للوحدة About كما هو موضح بالشكل التالى :

شكل توضيحى :



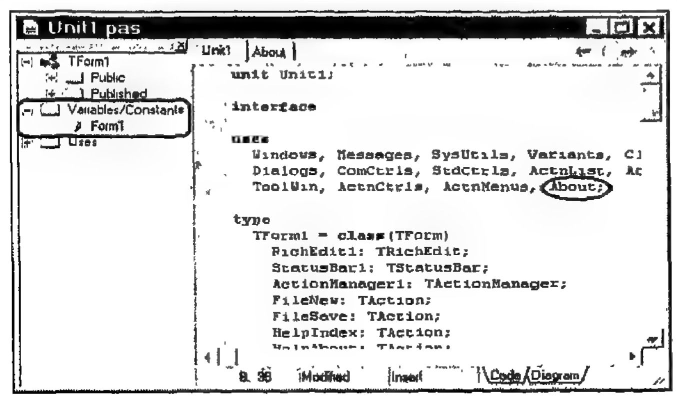




تلاحظ أن التبويب الخاص بالوحدة ActnRes غير موجود بمحرر الكود البرمجي وذلك لأنك لست في حاجة لهذه الوحدة.

(٧) انقر بالماوس على التبويب Unit1 ثم اضف الوحدة الجديدة About بإضافة الكلمة About إلى اسماء الوحدات الموجودة ضمن الجملة uses كما هو موضح بالشكل التالى :

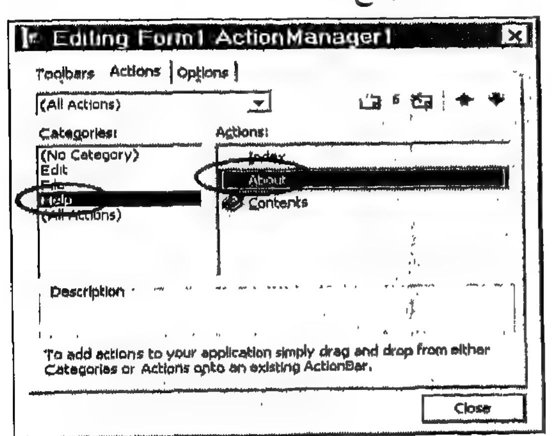
شكل توضيحي :



(٨) اضغط على المفتاح F12 بلوحة المفاتيح للعودة مرة أخرى إلى مود التصميم ثم انقر بالماوس نقرا مزدوجا على أيقونة المكون ActionManager لفتح صندوق الحوار Action Manager (إذا لم يكن مفتوح بالفعل).

(٩) انقر بالماوس على القسم Help بقائمة العرض Categories ثم على الفعل Index بقائمة العرض Actions كما هو موضح بالشكل التالى :

شكل توضيحي :



انقر بالماوس نقرا مزدوجا على الفعل About ليتم إنشاء أداة معاملة حدث خاصة به وذلك فى نافذة محرر الكود البرمجى Code Editor كما هو موضح بالشكل التالى :

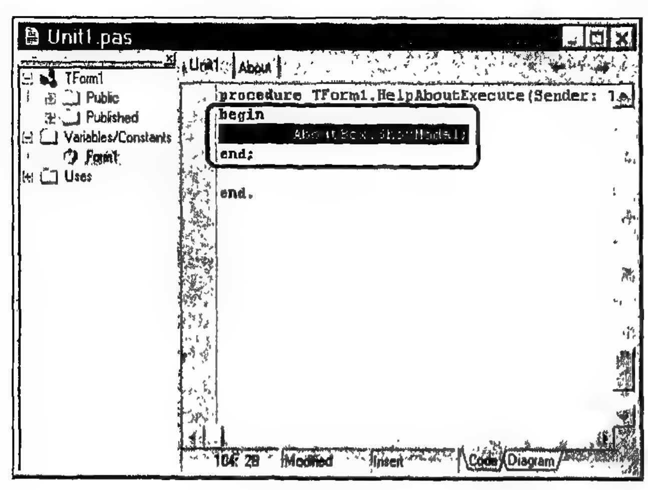
شكل توضيحى :

بالسطر الموجود به مؤشر الكتابة (بين الكلمة Begin والكلمة End) اكتب الكود التالى :

AboutBox.ShowModal;

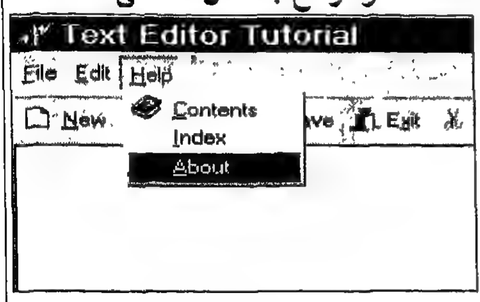
كما هو موضح بالشكل التالى :

شكل توضيحى :





هذا الكود يفتح صندوق الحوار About عندما يقوم المستخدم بفتح القائمة Help ويختار منها الأمر About كما هو موضح بالشكل التالـى :



استكمال التطبيق الذى تعده

التطبيق الذى تتولى إعداده أصبح مكتملا إلى حد كبير. ولكن على كل حال ستظل مطالب بتحديد وتوصيف بعض العناصر فى الفورمة الأساسية لهذا التطبيق.

لاستكمال هذا التطبيق اتبع الخطوات التالية :

(١) اضغط على المفتاح F12 بلوحة المفاتيح للتعامل مع الفورمة الأساسية بمود التصميم.

(٢) تـأكد مـن أن الفورمة نفسها هى المختاره وليس أحد العناصر الموجودة بها. ونود هنا القـول بأن القائمة المنسدلة الموجودة فى أعلى النافذة Object Inspector ينبغى أن تكون مشتملة على الاختيار Form1:TForm1 كما هو موضح بالشكل التالى :

شكل توضيحى :





إذا لم تكن هذه القائمة المنسدلة مشتملة على هذا الاختيار فى هذه الحالة افتحها واختر منها هذا الاختيار كما هو موضح بالشكل التالى :

Object Inspector

Form1	TForm1
FileNew	TAction
FileOpen1	TFileOpen
FileSave	TAction
FileSaveAs1	TFileSaveAs
Form1	TForm1
HelpAbout	TAction
HelpContents1	THelpContents
HelpIndex	TAction
Caption	Text Editor Tutorial

All shown

(٣) انقر بالماوس على التبويب Events ليظهر على السطح بالنافذة Object Inspector كما هو موضح بالشكل التالى :

شكل توضيحى :

Object Inspector

Form1 TForm1

Properties Events

Action
ActiveControl
Menu
ObjectMenuItem
OnActivate
OnCanResize
OnClick

All shown

افتح القائمة المنسدلة الخاصة بالحدث OnCreate ومنها اختر FormCreate وذلك لإنشاء أدة معاملة حدث يصف ما الذى يحدث عندما تم إنشاء الفورمة (منذ أن قمت بفتح التطبيق).

(٤) يظهر على الشاشة محرر الكود البرمجى Code Editor وفيه اكتب الكود التالى (بالسطر الموجود به مؤشر الكتابة بين Begin وEnd) :

```
Application.HelpFile := ExtractFilePath(Application.ExeName) +
'TextEditor.hlp';
FileNew.Execute
```

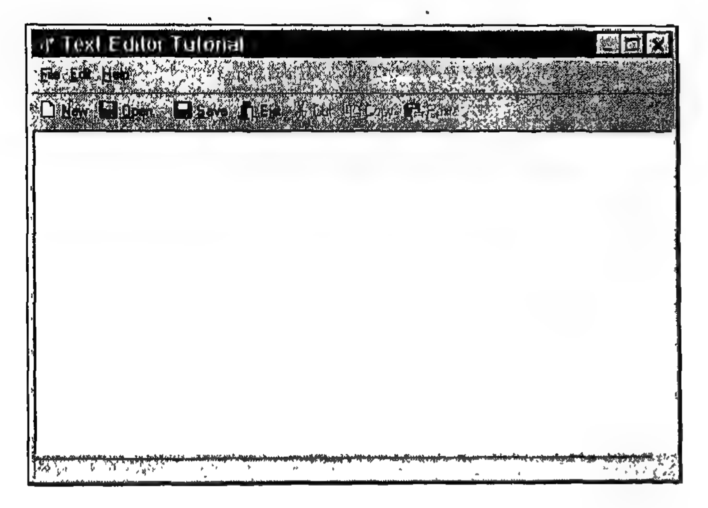
الهدف من هذا الكود الإعلان الابتدائى initializing عن التطبيق وذلك عن طريق كل من الآتى :

- إلحاق ملف مساعدة.
- جعل قيمة الخاصية FileName عبارة عن untitled.txt.
- وضع اسم الملف بسطر الحالة.
- تنظيف منطقة تحرير النص.

(٥) والآن افتح القائمة File ثم اختر منها الأمر Save All لتحفظ التغييرات التى أجريتها بالمشروع.

(٦) اضغط على المفتاح F9 بلوحة المفاتيح لتشغيل البرنامج الذى أصبح مكتملا الآن كما هو موضح بالشكل التالى :

شكل توضيحى :



الملحق

التحديث من الإصدارات السابقة
إلى Delphi 6



مقدمة عامة

عـندما تسـتخدم Delphi 6 لتحـميـل مشروع تم إعداده من خلال Delphi 5 أو من خـلال الإصـدارات الـتى تسـبقه مـن لغة Delphi فى هذه الحالة يتم تحديث مكونات هذا المشروع تلقائيا. ونحن فى هذا الفصل سنحاول سويا دراسة التغييرات التى تؤثر بشكل قوى ومباشـر عـلى المشـاريع الـتى تم إعدادها بالإصدارات السابقة من لغة Delphi عند التعامل معها من خلال Delphi 6. وهذه التغييرات يمكن حصرها فى النقاط التالية :

- التغييرات الخاصة ببيئة التطوير المتكاملة IDE.
- التحديثات التلقائية لأسماء الحزم البرمجية.
- التغييرات الخاصة بالتوافق مع الإصدارات السابقة.

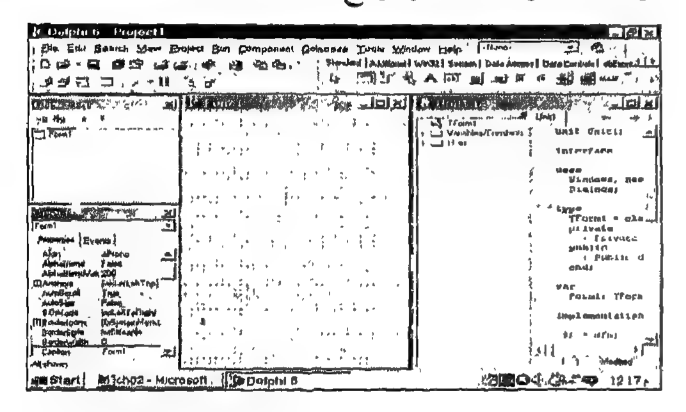


يمكنك الرجوع إلى الفصل الأول "الجديد فى Delphi 6" للحصول على المزيد من المعلومات حول المظاهر والإمكانيات الجديدة التى قد ترغب فى التعامل معها بالتطبيقات التى تتولى إعدادها.

التغييرات الخاصة ببيئة التطوير المتكاملة IDE

عندما تفتح المشروع الذى تتولى إعداده وكان هذا الفتح لأول مرة من خلال Delphi 6 فى هـذه الحالـة لـن يـبدو بنفس الشكل الذى كان يظهر به فى الإصدارات السابقة من لغة Delphi. فهـناك العديد من التغييرات التى تم إجراؤها على بيئة التطوير المتكاملة IDE بما فيها التغييرات التى أجريت على تخطيط النافذة وباللية المكونات كما هو موضح بالشكل التالى :

شكل توضيحى :





للمزيد من المعلومات حول هذه التغييرات يمكنك الرجوع لقسم المظاهر والإمكانيات الجديدة ببيئة التطوير المتكاملة IDE بالفصل الأول "الجديد فى Delphi 6".

التحديثات التلقائية لأسماء الحزم البرمجية

كما هو الحال فى التحديثات الأولية Prior للغة Delphi نجد أن اسماء العديد من الحزم البرمجية قد تغيرت. فعلى سبيل المثال قد تغير اسم الحزمة البرمجية vclide50 وأصبح vclide60. وفى هذا الصدد نقول إن الإصدارات السابقة للغة Delphi تقوم بأداء تحديثات تلقائية لأسم الحزمة البرمجية كذلك الحال بالنسبة هذا الإصدار الجديد من لغة Delphi والذى يقوم بنفس الشيء. ونود هنا القول بأن الآلية الخاصة بالتحديثات قد تغيرت تماما فهى الآن تعتمد بشكل أساسى على قائمة تضم مجموعة الملفات التى تغيرت. هذا ويمكن العثور على هذه القائمة فى الملف Delphi.upg الموجود فى الفهرس Bin والموضح فى الشكل التالى :

شكل توضيحى :

```
Delphi - Notepad
File  Edit  Search  Help
[Packages]
dss50=dss
ibevnt50=ibevnt
inet50=inet
inetdb50=inetdb
nmfast50=nmfast
qrpt50=qrpt
tee50=tee
teedb50=teedb
teeqr50=teeqr
teeui50=teeui
vcl50=vcl
vclado50=adortl
vclbde50=bdertl
vcldb50=vcldb
vcldbx50=vcldbx
vclib50=ibxpress
vclie50=vclie
vcljpg50=vcljpg
vclmid50=dsnap
vclsql50=dbexpress
vclsmp50=vclsmp
vclmcn50=dsnapcon
vclx50=vclx
webmid50=webdsnap
dbx50=dbx
dsnide50=designide
applet50=applet
coride50=coride
crbide50=crbide
dcl31w50=dcl31w
dclado50=dclado
dclaxserver50=dclaxserver
```



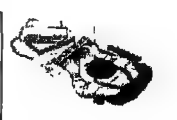
الطريقة السابقة تعتبر بديل لطريقة تغيير الـsuffixes.

فى الماضى عندما كانت لغة Delphi تشعر بأى خطأ يمكن أن يكون مرتبطا بتغيير فى اسم أى من الحزم البرمجية فى هذه الحالة تقوم بالبحث عن اسماء الحزم البرمجية

وذلك من أجل العثور على Obselect suffixes ثم تقوم بشكل تلقائى باستبدال ما تعثر عليه بالـSuffixes الصحيحة.

على سبيل المثال نجد أن الإصدار الخامس من لغة Delphi يقوم تلقائيا بتحديث الإشارة المرجعية للحزمة البرمجية Package40 (فى حالة وجود هذه الحزمة البرمجية) ومن ثم فإنه يقراء الحزمة Package50 بدلا منها.

الإصدار السادس من لغة Delphi يؤدى المهمة السالفة الذكر وذلك عن طريق البحث بقائمة تضم اسماء الحزمة البرمجية الـObselect ثم يقوم باستبدالهم كما هو مشار إليه بهذه القائمة.



هذه القائمة موجودة بالملف Delphi.upg الموجود بدوره فى الفهرس Bin داخل المجلد الذى تم تركيب لغة Delphi 6 فيه.

يستطيع مستخدمو التطبيق مشاهدة وتغيير محتويات الملف Delphi.upg باستخدام أى برنامج من برامج تحرير النصوص البسيطة بما فيها محرر النصوص الخاص ببيئة التطوير المتكاملة IDE للغة Delphi كما هو موضح بالشكل التالى :

شكل توضيحى :

```
C:\Program Files\Borland\Delphi6\Bin\Delphi.upg
Unit  Delphi.upg
[Packages]
dss50=dss
ibevnt50=ibevnt
inet50=inet
inetdb50=inetdb
nmfast50=nmfast
qrpt50=qrpt
tee50=tee
teedb50=teedb
teeqr50=teeqr
teeui50=teeui
vcl50=vcl
vclado50=adortl
vclbde50=bdertl
vcldb50=vcldb
vcldbx50=vcldbx
vclib50=ibxpress
vclie50=vclie
vcljpg50=vcljpg
vclmid50=dsnap
vclsql50=dbexpress
vclsmp50=vclsmp
vclmcn50=dsnapcon
```

ومثل هذه الإمكانية قد تكون مفيدة جدا وخصوصا لمصممى المكونات والكائنات وذلك لكونهم يرغبوا فى تعديل المكونات التى يقوموا بتصميمها بحيث تصبح متوافقة مع Delphi 6 والاحتفاظ بالمكونات فى حزمة برمجية مختلفة.

التغييرات الخاصة بالتوافق مع الإصدارات السابقة

فيما يلى نقط التوافق العامة التى قد تؤثر بشكل مباشر فى التطبيقات التى تعدها بلغة Delphi :

- مصدر توفير البيانات والأحداث المرتبطة بمجموعة بيانات العميل والتى تتأثر بالتغير فى البناء الهيكلى للقطاع VCL.
- التغيير المطلوب إجراؤه بالكود البرمجى من أجل صندوق حوار الولوج لقاعدة البيانات Database Login (الافتراضى).
- عدم التوافق بين ملفات البيانات الثنائية الخاصة بالفورم.
- التغيير فى الثوابت القابلة للكتابة Writeable.
- Unary negation لنوع البيانات Cardinal.
- إعادة تسمية DsgnIntf والتغييرات المرتبطة به.
- التغييرات التى أجريت على محرر المكونات.
- التغييرات التى أجريت على المكون TDesignWindow.
- التغييرات التى أجريت على الحزمة البرمجية VCL.
- انتقال وحدة الوسيط OpenGL إلى rtl.dcp.
- الأنواع التى انتقلت من الوحدة HTTPApp.pas إلى الوحدة HTTPProd.pas.
- إزالة وحدة البحث وإنتقال وتغيير SearchBuf.

هناك بعض الموضوعات الخاصة فى الفصل الأول "الجديد فى Delphi 6" قد تؤثر بشكل أساسى ومباشر فى توافق التطبيقات التى تعدها مع الإصدار ٦ للغة Delphi ومن بينها ما يلى :

- المظاهر الجديدة لمترجم اللغة.
- المظاهر الجديدة للقطاع VCL (على الأخص الجزء المسمى "التغييرات والوحدات المضافة").



مصدر توفير البيانات وأحداث مجموعة بيانات العميل المتأثرة بالتغيير في البناء الهيكلي للقطاع VCL

التقديم للمكون TCustomClientDataSet يتطلب إجراء تغييرات على أدوات معاملة الأحداث بالأكواد البرمجية التي تم كتابتها من خلال الإصدار الخامس من لغة Delphi والإصدارات السابقة له.

هناك ستة أحداث تنتمي إلى خمسة أنواع تأثرت بالتغيير التي أجرى على الوحدة DBCLIENT.PAS وأنواع هذه الأنواع والأحداث عبارة عن الآتي :

النوع	الحدث والتغيير
TResolveErrorEvent	يؤثر على الحدث OnUpDateError الخاص بمصدر توفير البيانات.
TBeforeUpdateRecordEvent	يؤثر على الحدث BeforeUpdateRecord الخاص بمصدر توفير البيانات.
TAfterUpdateRecordEvent	يؤثر على الحدث AfterUpdateRecord الخاص بمصدر توفير البيانات.
TProviderDataEvent	يؤثر على كل من الحدث OnGetData والحدث OnUpdateData الخاص بمصدر توفير البيانات.
TReconicleErrorEvent	يؤثر على الحدث OnReconcile الخاص بقاعدة بيانات العميل.



عند استخدام أدوات المعاملة الخاصة بالأحداث المذكورة في الجدول السابق ينبغي عليك استبدال TClientDataSet بـTCustomClientDataSet.

التغيير المطلوب إجراؤه بالكود البرمجي لصندوق الحوار الافتراضي Database Login

فـي السـابق كـان تحديـد قـيمة الخاصـية LoginPrompt لأى مكون من مكونات الاتصـــال (مـــثل المكـــون TDatebase أو المكـــون TADOConnection والمكـــون TDCOMConnection) يجعـل صـندوق الحوار الافتراضى Database Login يظهر على الشاشـة. ولكـن الآن لم يعـد هـذا الـتحديد يؤدى إلى ظهور صندوق الحوار هذا وذلك إذا لم تقـوم بإضـافة DBLogDlg إلى الجملة Uses بالكود البرمجى الذى تكتبه. وفى هذا الصدد نقـول إن التطبـيقات الـتى تعـتمد على صندوق الحوار الافتراضى Database Login يجب تعديـل الكـود الـبرمجى الخـاص بها لإضافة DBLogDlg إلى الجملة Uses وإلا فلن تقوم هذه التطبيقات بعرض صندوق الحوار هذا والذى يطلب اسم المستخدم وكلمة المرور.

عدم التوافق بين ملفات البيانات الثنائية الخاصة بالفورم

فـى الماضـى كانـت الإصـدارات الأقـدم مـن لغـة Delphi لديهـا القدرة على قراءة الملفـات الثنائـية للفـورم (أو الملفـات DFM) الـتى يتم إنشاؤها بالإصدارات الأجدد من لغة Delphi. ولكـن هذه الإمكانية لم تعد متاحة فى Delphi 6 فهناك بعض ملفات الفورم الثنائية الـتى يمكن قرائتها بطريقة غير صحيحة وذلك بسبب الطريقة التى يتبعها Delphi 6 لأداء عملية الـStreaming الداخلية للسلاسل الحرفية.

هـذا وفى الماضى كانت عملية الـStreaming تتم مفترضة مجموعة حرفيات محلية معيـنة. ولكـن الآن نجـد أن عملية الـStreaming تفترض أن مجموعة الحرفيات عبارة عن UTF-8. ومـن ثـم لـو كانـت هـناك حرفـيات مكودة بأكواد أكبر من ١٢٧ (مثل رمز حقوق الاسـتخدام ©) فـى الملف الثنائى للفورمة المعدة بلغة Delphi 6 فإم مثل هذا الملف لن تتم قراءته بواسطة الإصدارات الأقدم للغة Delphi.



لو أنك تنوى أن تستخدم ملف فورمة معدة بـDelphi 6 (أو ملفات فورم معدة بإصدارات سابقة للغة Delphi تم إستيرادها وتعديلها من خلال Delphi 6) فى اصدار أقدم من لغة Delphi فى هذه الحالة ينبغى حفظ هذا الملف بتنسيق نصى بدلا من التنسيق الثنائى.

التغيير في الثوابت القابلة للكتابة Writeable

الحالة الافتراضية لمحول مترجم اللغة $WRITEABLECONST (aka $J) أصبحت الآن OFF مما سيؤدى إلى منع المشاريع المعدة بلغة Delphi من أن يكون لديها ثوابت من النوع Writeable. وهذه النوعية من الثوابت تشير إلى استخدام لثابت محدد النوع على أساس كونه متغير قابل للتعديل فى مرحلة التشغيل run-time. وفيما يلى مثال لذلك :

```
const
        foo: Integer = 12;
begin
                foo := 14;
end.
```

فى الإصدارات السابقة للغة Delphi كان يعد ذلك Construct مقبولا ولم تكن الثوابت فى حقيقية الأمر ثوابت. ولكن مع كون المحول $WRITEABLECONST لا يعمل OFF فإن هذا الكود لن يجعل أى مترجم يعطى رسالة خطأ عند تعامله مع جملة التخصيص للمتغير foo فى البلوك Begin...End. ولإصلاح هذا عليك أن تقوم بتغيير الإعلان عن الثابت ليصبح إعلان عن متغير var.

من الممكن أن يكون لديك كود برمجى يستخدم النوع Const للإعلان المبدئى عن متغير محلى بعمر افتراضى عام global lifetime كما هو موضح بالكود البرمجى التالى :

```
procedure MyProc;
const
        somedata: Integer = 12;
begin
        Inc(somedata, 3);
end;
```

ستحتاج لأن تنقل المتغير الذى ينتمى للنوع const إلى خارج الإجراء والإعلان عنه على أساس أنه متغير عام. وبعد إجراء هذا التغيير سيصبح الكود البرمجى السابق كالآتى :

```
var
        somedata: Integer = 12;
```

```
procedure MyProc;
begin
        Inc(somedata, 3);
end ;
```

عندما يكون الكود البرمجي معتمدا بشدة على النوع Const Quirk (مثل أداة تكوين أدوات التحكم ActiveX) في هذه الحالة يمكن إدراج الموجهة {$WRITEABLECONST ON} بملف المصدر المشتمل على الكود البرمجي على أساس أن ذلك حل سريع. ومثل هذا العمل محظور في كل من RTL وVCL وCLX والكود البرمجي المصدري لقواعدة البيانات DB كما إنه يعتبر عمل غير مشجع في المصادر IDE ولكنه مقبول بالنسبة للوحدات الثانوية مثل الـ Wrappers الخاصة بأدوات التحكم ActiveX.

بصفة عامة ينبغي عليك ملاحظة أن عبارة "ثابت قابل للكتابة Writeable constant" ما هي إلا إجتماع لكلمتين على طرفي نقيض تماما. فالإصدارات السابقة للغة Delphi كانت لهم بشكل افتراضي للإبقاء على مستوى التوافق ثابت بدون أى خلل مع النسخ القيمة من مترجم اللغة والتي كانت تتعامل مع أنظمة التشغيل الـ16Bit لم تعد مهمة لأغلب المبرمجين بلغة Delphi.



ننصحك دوما بأن تتبع أسلوب برمجي جيد وحاول أن تتفادى الثوابت التي من النوع Writeable.

Unary negation لنوع البيانات Cardinal

في الماضي كانت لغة Delphi تتعامل مع Unary negation لأرقام تنتمي لنوع البيانات Cardinal وذلك باستخدام معاملات التشغيل الخاصة بأنظمة التشغيل الـ٣٢ بت مما يؤدى إلى الحصول على بعض النتائج الفردية.

هنا مثال لكود برمجي يستخدم Unary negation :

```
var
        c: Cardinal;
        i: Int64;
begin
```

```
        c := 4294967294;
        i := -c;
        WriteLn(i);
end ;
```

فـى الإصدارات السابقة للغة Delphi نجد أن قيمة i التى يتم عرضها عبارة عن ٢ وهـذا مـن الواضـح سـلوك غـير صحيح تماما بالنسبة لهذه الحالة. ولكن فى Delphi 6 يتم التعامل مع Unary negation بعد السماح للنوع Cardinal بأم يكون نوع له إشارة ٦٤ بت ومن ثم القيمة النهائية لـ i والتى يتم عرضها تكون 4294967294-.

إعادة تسمية DsgnIntf والتغييرات المرتبطة به

الإشارات المرجعية إلى DsgnIntf فى المشروع الذى تتولى إعداده يجب أن تتغيرت إلى الأسم الجديد DesignIntf بالإصدار السادس للغلة Delphi. كذلك قد تحتاج إلى إضافة DesignEditors وVCLEditors وRTLConsts بالجملـة Uses بـالكود الـبرمجى الـذى تكتـبه. كذلـك سـتحتاج لأن تضيف DesignIde إلى قائمة متطلبات الحزمة البرمجية التى تتولى إعدادها. أما الإشارات المرجعية References إلى dsnide50 فينبغى أيضا تغييرها إلى DesignIde لو أن هذا التغيير لم يتم تلقائيا بواسطة لغة Delphi نفسها.

أى حـزمة برمجـية خاصـة بمـرحلة التشـغيل runtime packages تسـتخدم IDesigner تحـتاج لأن تسـتخدم IDesignerHook لكـى تتفادى متطلبات الـdesignide فـى مـرحلة التشـغيل. هـذا وفـى الكـود الـبرمجى لمـرحلة التشـغيل يجـب أن يكـون IDesignerHook وافـيا للغـرض. أمـا الكـود الـبرمجى لمرحلة التصميم فيمكن أن يستخدم IDesigner ولكن ينبغى أن يستخدم شيء ما يشبه الآتى :

```
var
        RealDesigner: IDesigner;
...
        SomeDesignerHook.QueryInterface(IDesigner,RealDesigner);
...
```

وذلـك للحصول على واجهة استخدام (وسيط) IDesigner حقيقية من أى حالة من حالات الـــIDesignerHook يتطلب فقط قطاعات وفورم لكى يكون متاحا للاستخدام. هذا ويحتاج

الـIDesigner إلى DesignIntf والذى يتضمن العديد من الحزم البرمجية الأخرى بعض منها قد يكون غير قابل لإعادة النشر والتوزيع مرة أخرى.

التغييرات التى أجريت على محرر المكونات

القطاع TComponentEditor لدية سلسلة نسب مختلفة فى Delphi 6. ففى Delphi 5 كان هذا القطاع ينحدر من القطاع TInterfaceObject ولكنه الآن ينحدر من القطاع الجديد TBaseComponentEditor. كذلك فإن القطاع TComponentEditorClass أصبح الآن قطاع ينحدر من TBaseComponentEditor وذلك بدلا من أن ينحدر من القطاع TComponentEditor. وهذه التغييرات فى البناء الهيكلى قد تتطلب منك أن تقوم بتعديل المشاريع التى صممتها من خلال الإصدارات السابقة للغة Delphi.

التغييرات التى أجريت على القطاع TDesignWindow

لقد تم إجراء عدد من التغييرات المرتبطة بالقطاع TDesignWindow. فلقد تم نقله إلى الوحدة DesignWindows كما أن الأسلوب FormClosed الخاص به قد تن استبداله بالأسلوب الجديد DesignerClosed. وفى الماضى كان أى شخص يستطيع الوصول إلى الفورمة من خلال الأسلوب FormClosed وذلك باستخدام المعامل AForm الخاص بهذا الأسلوب. ولكن مع الأسلوب الجديد DesignerClosed أصبح من الضرورى الآن استخدام الخاصية Root الخاصة بـDesigner للوصول إلى الفورمة.

من خلال الأسلوب FormClosed كان أى شخص يستطيع إنشاء قوائم اختيار عن طريق استدعاء الخاصية Create للمكون TDesignerSelectionList أو الخاصية Create للمكون TComponentList. أما الآن فلكى يتم إنشاء قوائم اختيار من خلال الأسلوب الجديد DesignerClosed فمن الضرورى ان يتم استخدام وسيط IDesignerSelections. وأنت تستطيع إنشاء واحد من هذه الأوساط باستخدام الدالة CreateSelectionList.



المعاملات الخاصة بالأسلوب SelectionClosed أصبحت الآن تختلف عما كانت علية فى Delphi 5.



التغييرات التي أجريت على الحزم البرمجية VCL

محتويات بعض الحزم البرمجية المرتبطة بـVCL قد تم إعادة توزيعها من خلال حزم برمجية أخرى. ولو أنك قمت بإعداد إشارات مرجعية إلى الوحدة vcl50.dcp بالمشروع الذى تتولى إعداده فى هذه الحالة ستحتاج لتغيير هذه الإشارات المرجعية إلى وحدات أخرى مثل الوحدة vcl.dcp والوحدة rtl.dcp.

انتقال وحدة الوسيط OpenGL إلى rtl.dcp

وحدة الوسيط OpenGL التى أعددتها شركة Borland (الموجودة بالملف Opengl.dcu) كانت فى الماضى وحدة مستقلة بذاتها فى Delphi 5 وكانت موجودة بالمجلد Lib. ولكن الآن فقد تم ضمها داخل الملف rtl.dcp وذلك فى Delphi 6. ومثل هذا التغيير قد يتسبب فى حدوث بعض المشاكل بالنسبة للمشاريع التى تم إعدادها من خلال Delphi 5 والتى يتم فتحها وتشغيلها من داخل Delphi 6.

لتوضيح ما سبق سنذكر المثال التالى :

فى مشروع معد بـ Delphi 5 كان من الممكن عمل تخطى بطريقة معينة للوحدة OpenGL وذلك عن طريق وضع وحدة تكون بنفس الأسم فى مكان ما بالمسار الخاص بالمشروع. ولكن عند استخدام نفس الطريقة فى Delphi 6 فإن ذلك يتسبب فى حدوث الخطأ name conflict فى أى مكون يستخدم rtl.dcp ومن ثم يكون من المطلوب الآن تغييز الاسم.

الأنواع التى انتقلت من الوحدة HTTPApp.pas إلى الوحدة HTTPProd.pas

لقد تم نقل العديد من الأنواع الموجودة فى الوحدة HTTPApp لتصبح الآن فى الوحدة HTTPProd. وهذه الأنواع عبارة عن THTMLBgColor وTHTMLAlign و THTMLVAlign. ولو كانت المشاريع التى تعدها تستخدم أى من هذه الوحدات فى هذه الحالة ينبغى عليك تغيير الجمل uses الموجودة بالأكواد البرمجية لهذه المشاريع بحيث تشير إلى الوحدة HTTPProd بدلا من الوحدة HTTPApp.

إزالة وحدة البحث وإنتقال وتغيير الروتين SearchBuf

لم تعد الوحدة Search موجودة فى Delphi 6. أما بالنسبة للروتين SearchBuf والذى يعمل على تحديد موضع سلسلة حرفية فرعية داخل مخزن للنصوص Text Buffer قد تم نقله إلى الوحدة StrUtils كما أن المعاملات الخاصة بهذا الروتين قد تغيرت هى الأخرى. والمعامل النهائى أصبح الآن الكائن TStringSearchOptions. ولو أن المشروع الذى تتولى إعداده لن تتم ترجمته من خلال Delphi 6 بسبب أن المترجم لا يستطيع العثور على الوحدة Search فى هذه الحالة قم بتغيير الجملة uses بحيث تضم StrUtils بدلا من Search. كما إنك ستحتاج أيضا فى مراجعة الاستدعاءات التى أعددتها إلى SearchBuf وذلك لكى تتأكد من أنك المعاملات التى تستخدمها تتطابق مع الصياغة الجديدة.

فهرس الكتاب

لقد

أعددنا هذا الكتاب من أجـل أن

نتوجه به إلى المبرمجين عامة المبتدئين منهم خاصة الذين

يخطون خطواتهم الأولى فى مجال البرمجة من خلال الإصدار السادس للغة Delphi.

هذا ونود هنا القول بأن هناك فجوة كبيرة بـين الإصدارات السابقـة للغة Delphi وهـ

الإصدار الجديد الذى جعل لغة Delphi بحق لديها القدرة على منافسة أقوى لغات البرمجـ

التى تنتمى للطائفة Object Oriented Programming (OOP) مثل لغة ++C و #C

و ++VC ولغة Java

من خلال Delphi 6 يمكننا الآن إعداد تطبيقات قـواعد بـيانات تعمل فى

شبكات الحاسب الآلى الضخمة بالإضافة إلى إمكانية إعداد التطبيقات التى تعمل

بخوادم شبكة الويب فضلا عن إمكانية إعداد التطبيقات المكتبـية Desktop

Application وتطبيقات الموزعة Distributed Applications

وتطبيقات الوسائط المتعددة.لقد راعينا أن تكون طريقة الشرح سهلة

وبسيطة ومدعمة بالكثير من الأشـكال التوضيحـية والأمثلة

العملية لكى تتناسـب مع مسـتوى المبـرمجين جميعا

وخصوصا المبتدئين

والله الموفق

الناشر



دار الكتب العلمية للنشر والتوزيع

٥٠ شارع الشيخ ريحان - عابدين - القاهرة

☎ ٧٩٥٤٢٢٩

I.S.B.N 977-287-277-3